Acknowledgements

I express my deep gratitude to my guides Prof. S.M. Deshpande and Dr S.S. Desai for suggesting this problem, for their inspiring guidance through out the course of this research work, and for the patient discussions they had with me at difficult times during the course of the work.

Research is not the kind of work where there is a well defined set of instructions to be carried out. It is more like getting into a thick jungle or diving into a deep sea in hope of discovering some valuable hidden treasures. At times especially making a 3-D code work which is written from the scratch is like a venture described above. There were times when the work would not progress for months. I was able to wriggle out of such difficulties only by the constant support and encouragement given by Prof. S.M Deshpande and Dr S.S. Desai. In fact it is only due to the homely environment provided by Dr S.S. Desai at my office, that enabled me to complete my thesis without any sort of mental worries.

I take this opportunity to express my thanks to Dr. P.S.Kulkarni of CFD lab, for his ready assitance on many ocassions at the CFD lab which made my visits to CFD lab a pleasent experience.

Any CFD code developer knows that, writing a code takes a day, but it takes an year to debug the code to make it work satisfactorily. I thank Dr. A.K. Ghosh of ADA who has helped me in not only discussing how to go about in writing the code but also sat with me for days in debugging my 3-D codes. I wish to express my gratitude to Ghosh(father of LSKUM) for explaining me the details of LSKUM on various occasions. I would be failing in my duties if I would not mention the name of my good friend Mr. K. Anandhanarayanan, who is also student of my guide Prof. Deshpande. I have immensly derived a lot of tips from him especially by the discussions we had on quadtree implementation.

Many a times it is said, behind every success there is a person behind the scene. I think this person in my case is Dr J.S. Mathur who is my collegue at NAL. At times of difficulty he would patiently hear me and give suggetions. In fact all my grid generation needs were readily offered by him. Preperation of the thesis in LaTex was a very easy job for me. At every stage I had to only turn back and ask 'Mathur what should I do next !'. I wish to express thanks for all the help derived from him.

Finally my sincere thanks to my parents and my wife Parimala for having provided me with the right environment at home, without which it would not have been possible to carry out this work.

List of Authors publications

Publications in Journals

1. Ramesh,V. and Deshpande, S.M., *Least Squares Kinetic Upwind Method on moving grids for unsteady Euler computations*, Computers and Fluids Journal, Vol. 30/5, pp. 621-641, May 2001.

2. Dauhoo, M.Z., Ghosh, A.K., Ramesh, V. and Deshpande, S.M., *q-LSKUM : A new higher order kinetic upwind method for Euler equations using entropy variables*, Computational Fluid Dynamics Journal, Vol. 9, No. 1, April 2000.

Papers in Proceedings/Conferences/Symposia etc.

1. Ramesh, V. and Mathur, J. S., *A comparison of upwind schemes on unstructured grids*, Proceedings, 7th Asian Congress of Fluid Mechanics, Chennai, Vol. 2, pp. 609–612, December 8–12, 1997

2. Balasubramanian, R., Ramesh, V. and Deshpande, S.M., *On Kutta-Joukowski condition*, Proceedings, 7th Asian Congress of Fluid Mechanics, Chennai, Vol. 1, pp. 287-290, December 8–12, 1997

3. Dauhoo, M.Z., Raghurama Rao, S.V., Ramesh, V. and Deshpande, S.M., *A strong implementation of kutta-Joukowski condition using peculiar velocity based upwind method*, Proceedings 16th International Conference on Numerical Methods in Fluid Dynamics, Arcachon, France, July 6–10, 1998, Lecture Notes in Physics, Vol. 515, pp. 367–372, Springer, 1998.

4. Anandhanarayanan, K., Dhokrikar, D.B., Ramesh, V. and Deshpande, S.M., *A grid free method for 2-D Euler computations using least squares kinetic upwind method*, Proceedings 3rd Asian Computational Fluid Dynamics Conference, Vol. 2, pp. 390–395, Dec 7–11, 1998, Bangalore, India.

5. Deshpande, S.M., Ghosh, A.K., Ramesh, V. and Dauhoo, M.Z., *New developments in Least Squares Kinetic Upwind Method for Euler Equations*, 8th International conference on Hyperbolic problems Theory-Numerics-Applications, Otto-von-Guericke-Universitat Magdeburg, Germany, Feb. 28- Mar.3, 2000.

6. Dauhoo, M.Z., Ghosh, A.K., Ramesh, V. and Deshpande, S.M., *q-LSKUM - A New higher order kinetic upwind method for Euler equations using entropy variables*, International Symposium on Computational Fluid Dynamics, Sep. 5-10, 1999, Bremen, Germany.

7. Ramesh, V. and Deshpande, S.M., *Euler computations on arbitrary grids using LSKUM* 1st International conference on Computational Fluid Dynamics, Kyoto Japan, July 10- 14, 2000

8. Jain Sachin, Ramesh, V. and Deshpande, S.M., *Least squares kinetic upwind method on moving grid for unsteady Euler computations*, 1st International conference on Computational Fluid Dynamics, Kyoto Japan, July 10- 14, 2000

9. Deshpande, S.M. and Ramesh, V.,*Meshless methods based on least squares kinetic upwind method*, GAMM 2000 workshop, 24-25 Nov. 2000, TU Braunschweig, Germany.

10. Deshpande, S.M., Anandhanarayanan,K., Praveen, C. and Ramesh, V.,*Theory and Application of 3-D LSKUM Based on Entropy Variables*, First ICFD, Oxford, March 2001.

Internal Reports

1. Ramesh, V., Mathur, J. S., and Deshpande, S. M., *Kinetic treatment of the far-field boundary condition*, Fluid Mechanics Report 97 FM 2, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, April 1997.

2. Ramesh, V., Ghosh, A.K., and Deshpande, S. M., *Computation of three dimensional inviscid compressible flows using least squares kinetic upwind method* , Fluid Mechanics Report 97 FM 8, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, Sept. 1997.

3. Ramesh, V. and Deshpande, S. M., *Least squares kinetic upwind method on moving grids*, Fluid Mechanics Report 99 FM 1, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, Jan. 1999.

Nomenclature

$\|A\|$	Matrix to transform static fluxes to moving fluxes
c_1, c_2	cartesian components of the peculiar velocity
C_A	axial force coefficient in 3-D
C_N	normal force coefficient in 3-D
C_M	pitching moment coefficient in 3-D
C_d	coefficient of drag in 2-D
C_l	coefficient of lift in 2-D
C_p	coefficient of pressure
c_v	specific heat coefficient for constant volume
e	internal energy per unit mass
erf	error function
F	Maxwellian velocity distribution function
f	velocity distribution function
GX, GY, GZ	Cartesian components of flux vector
$GX^{\pm}, GY^{\pm}, GZ^{\pm}$	Cartesian components of split flux vector in 3-D
$GX^{I,II,III,IV}$	2-D quadrant split x-component of flux vector in the four quadrants
$GY^{I,II,III,IV}$	2-D quadrant split y-component of flux vector in the four quadrants
$GX^{\pm\cdot\pm}$	x-component of quadrant split flux vector in 3-D
$GY^{\cdot\pm\pm}$	y -component of quadrant split flux vector in 3-D
I	internal energy variable
I_o	internal energy due to non-translational degrees of freedom
J	collision term in Boltzmann equation
M	Mach number
n	normal direction in the natural coordinate system
p	pressure
q	total velocity
R	gas constant
s	tangential direction in the natural coordinate system
Δs	total distance between any two points
T	temperature
t	time

Δt	time step
U	vector of conserved variables
u_p	piston velocity
u_1, u_2, u_3	Cartesian components of fluid velocity
$\bar{u_1}, \bar{u_2}$	Cartesian components of fluid velocity relative to the grid point velocitiy
v_1, v_2, v_3	Cartesian components of molecular velocity
$\bar{v_1}, \bar{v_2}$	Cartesian components of molecular velocity relative to the grid point velocitiy
w_1, w_2	Cartesian components of grid point velocity, also weighting functions
$\Delta x, \Delta y, \Delta z$	x,y,z distance between two points
α	angle of incidence
β	$1/2RT$
γ	ratio of specific heats
ρ	density of fluid
Ψ	moment function vector
∞	subscript denoting free-stream conditions

Abstract

Least Squares Kinetic Upwind Method(LSKUM) is a node based scheme. This is a kinetic theory based upwind scheme derived from the Boltzmann equation. The method makes use of the fact that, Euler equations are suitable moments of the Boltzmann equation when the velocity distribution function is a Maxwellian. LSKUM has the potential to operate on any type of mesh structured,unstructured,Cartesian,Hybrid, Overlapping(Chimera) meshes, or even an arbitrary distribution of points. Given any distribution of points(which can be obtained from any type of grid generator), the LSKUM solver needs the data of the immediate surrounding nodes at every point. This data of the immediate neighbouring points is called as the connectivity data or simply the connectivity.

A simple approach to build connectivity for the nodes, would be to search through all the points in the data set and then retain only those which lie within a small radius which defines the immediate neighbourhood of the nodes. When the number of points become more, a lot of time is unneccesarily spent in search operations. Therefore it is desirable to use suitable data structures to optimise the search operations. Towards this we have used quadtree data structures to build the connectivity data. Generation of connectivity is basically a geometric proximity problem. The quadtree data structures are well known for their applications to geometric proximity problems. In our present work we have developed a preprocessor which uses the quadtree data structures to generate the connectivity data. Given any arbitrary distribution of points, the preprocessor builds a background quadtree data structure around these points. Then using this quadtree structure the desired connectivity data is generated. The solver then operates on the given distribution of points with the connectivity data generated by the preprocessor.

The main advantage of this approach is, for any given distribution of points the solver can operate on them without the need to tesselete the points to form lines along any coordinate directions or to form any edges defining cells. For complex configurations, with such an approach we can use any simple grid generator to generate grids over each individual component. The final distribution of points is then obtained by combining all the nodes from each component grid. The preprocessor will then operate on these points to generate the connectivity data, thus facilitating the solver to operate. With this approach, we can also use overlapping grids without any need to keep track of the overlapped regions. In the conventional methods

for example using finite volume methods, the solver has to keep track of the various overlapped boundaries, also construct accurate interpolation schemes for transfer of solution across these various overlapped boundaries. In our present work using LSKUM in conjunction with the quadtree preprocessor, we do not face any of these difficulties. The solver only sees a distribution of points, at which the solution is directly updated. In fact in the overlapped regions we have more number of points at which the solution is directly updated. Hence we get a better resolution of the flow compared to the interpolation procedures which do not solve for any fluid dynamic equations in the overlapped regions.

In the first chapter we present the details of formulation for LSKUM. This includes the extension of LSKUM for 3-D problems which is a recent addition from the existing 2-D LSKUM. Along with the extension of LSKUM to 3-D, we also give a new kinetic theory based boundary treatment for implementing the far field boundary condition. This is called as the Kinetic Outer Boundary Condition(KOBC). This treatment significantly differs in its approach from the Riemann treatment for the outer boundary condition. KOBC is very similar for both 2-D and 3-D problems and unless like the Riemann treatment does not assume a local 1-D flow model normal to the outer boundary. Another important feature of KOBC, which makes it different from the Riemann treatment is, KOBC is independent of the flow conditions at the outer boundary. That is the formulae for KOBC remain identically same for all subsonic inflow/outflow and supersonic inflow/outflow conditions at the outer boundary. We have successfully used KOBC in all our 2D and 3D computations.

In the thesis we present results for 2-D LSKUM on a variety of point distributions conclusively establishing the grid free nature of the LSKUM solver. This has been made possible by the use of the quadtree preprocessor. Results are presented for flow past NACA0012 airfoil and NACA 0012 biplane configuration for various test conditions. The flow computations have been done using point distributions obtained from structured, unstructured, Cartesian and overlapping grids.

In order to establish the working of 3D LSKUM we have tested the solver on variety of problems. The first problem considered is the simulation of an intense blast wave in air. This problem deals with extreme flow conditions involving very high temperature gradients. This a problem of spherical symmerty with the spherical blast wave propagating radially outwards in time. In our computations we have used points from a simple Cartesian grid. In spite of using points from rectangular grid the method captures the shock front very well maintaining the spherical symmetry. In the next test case, we consider supersonic flow past a hemisphere. Through these two test cases we first establish the proper working of the interior and the boundary schemes. In order to further prove the working of the method, we compute transonic flow past ONERA-M6 wing. This is a more realistic 3D configurartion. The computed results compare very favourably with the AGARD values. Having obtained good results for the various test cases, we finally compute high supersonic flow past a generic flight vehicle. This is a fairly complex configuration consisting of Bluntcone-cylinder-flare with lifting surfaces. Extensive computations for different Mach numbers and angle

of attack have been carried out for this geometry. All the computations compare satisfactorily with the available experimental values.

In the end, we present two further extensions to LSKUM. The first case consists of extension of LSKUM to moving grids. This is called as LSKUM-MG. In the second extension we give a new novel approach for obtaining higher order schemes for LSKUM. We call this as LSKUM-NH (LSKUM-Novel Higher order). Formulation for 1D LSKUM-MG is followed by results on the standard moving piston problem. The formulation for 2D LSKUM-MG is then given along with the kinetic theory based boundary conditions for the moving boundaries. Results for flow past an airfoil oscillating in pitch is presented using the 2D LSKUM-MG. Finally we give some preliminary results for 1-D LSKUM-NH for 1-D shock tube problem.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

CFD has evolved as one of the powerful analysis and design tools over the past two decades. It was during 1970-80, CFD first began to have a significant impact on aerodynamics through the introduction of panel methods[23, 59, 60]. The panel code could solve the linear flow models for arbitrarily complex geometry in both subsonic and mildly supersonic flow. Satisfactory methods for treating the non-linear equations of transonic flow were later developed through the use of full potential equations[34, 36, 12, 13]. With the allowance for the boundary layer effects, the full potential methods are useful for 2-D flows and simple wing-body flows in transonic regime under attached flow conditions. For mildly separated conditions, the inviscid-viscous interaction method has also been used[70, 14]. However they are applicable only upto a local shock Mach number of 1.3-1.4. Beyond this, usually the rotational effects are not negligible and the potential methods are not valid. Euler equations are more suitable for flow dominated by non-linear and vortical flows but not much by viscous effects. Euler equations can therefore capture of strong shocks appearing in the flow fields. Euler equations have now become an industry tool for inviscid analysis[37, 16, 17, 71].

Numerous methods exist today for the solution of Euler equations. All the approaches invariably need meshes, which are used to divide the flow region into discrete subdomains. The discretization procedure must allow for the treatment of complex configurations. The principle alternatives are body fitted curvilinear meshes[16, 17], Unstructured tetrahedral meshes[81] and Cartesian meshes[27, 28] which are gaining popularity in the recent times. Each of these approaches have their own advatages as well as drawbacks. Body fitted meshes have been widely used. The commonly used procedures for mesh generation are algebriac transformations[8, 24], methods based on solution of elliptic equations, pioneered by Thompson[75], and methods based on the solution of hyperbolic equations marching out from the body[72]. In order to treat very complex configurations the general practice is to use multiblock[17, 78] procedure(also called block-structured grid), with separately generated meshes in each block, which may then be patched at the block faces. When these are allowed to overlap we obtain Chimera meshes[6, 9]. In practice the generation of satisfactory

multiblock structured grid for a complex configurartion may require several months of effort[40].

The alternative is to use an unstructured mesh in which the domain is subdivided into tetrahedra using Delaunay triangulation[7, 79] or moving front method[47]. This approach is also fast gaining popularity. Also in recent times Cartesian meshes are being used increasingly in many Euler applications[10, 11, 27, 28, 52, 55, 58, 62]. Development of Cartesian grids in the region away from a body is much more easier. But these methods generally lose accuracy near curved boundaries. This difficulty may be alleviated by using adaptive mesh refinement[11].

Associated with the choice of mesh type is the method of the discretization procedure for the Euler equations. Finite difference, Finite Volume and Finite element methods are the main types used for the discretization of the Euler equations. Finite difference methods are usually used for structured grids. Unstructured grids are used for finite element methods as well as finite volume methods. Finite volume methods can be used for many types of grids including structured as well as unstructured grids. Cartesian meshes are generally used in an unstructured grid format using finite volume approach.

Again the solution procedures or the algorithms used for Euler equations are basically of two types. In the first type, discretization procedures lead to non-dissipative approximations to Euler equations. Artificial dissipative terms are then added(so called addition of controlled diffusion) to stabilise the solution and to capture shock waves and contact discontinuities without undesirable oscillations. This approach has been succesfully used in many applications[16, 17, 38, 39]. In the second type following the pioneering work of Godunov[35], a variety of dissipative and upwind schemes have been developed[61]. Recently there has been an increasing interest in Kinetic Flux Vector Splitting(KFVS) schemes which are based on the Boltzmann equation appearing in the kinetic theory of gases[15, 18, 19, 20]. In our present work we follow the approach developed by Deshpande[18, 19, 20].

1.1 Motivation for the present work

The group of Deshpande [18, 19, 20, 49, 50, 51, 63, 64, 65] at CFD lab, IISc has developed and applied these kinetic upwind schemes to a wide range of practical problems. The famous Kinetic Upwind Flux Vector(KFVS) scheme was first developed here by Deshpande and Mandal[18, 49, 50, 51]. Several co-workers have further developed and applied this method for different practical problems. Mathur[53, 54, 55] has shown KFVS to work on different types of meshes for a variety of problems in a finite volume frame work. Of the many upwind schemes available for the solution of Euler equations, Villedieu[25] has shown that KFVS is one of the best schemes for Euler equations with respect to properties like postivity preservation for second order calculations. Also Mathur[56], Anand et. al.[5] and Lou. et.al. [46] have also extended the KFVS schemes for Navier Stokes equations. The CFD group at DRDL[22, 71] infact

have developed BHEEMA code based on the 3-D cell centered KFVS finite volume method and applied to a number of practical problems. A node based scheme using KFVS was first developed by Ghosh[29, 31, 21]. This method called Least Squares Kinetic Upwind Method(LSKUM) has been shown to work for 2-D flows.

It is a well known fact that, generation of a suitable grid for a complex geometry involves several months of work. As already discussed there are several ways of generating meshes. For a structured grid implementation of higher order schemes are relatively easy compared to implementation on tetrahedral unstructured grids. Unstructured grids generally need reconstruction procedure to obtain higher order schemes which usually come at an expense of more computational time. But these tetrahedral meshes are more easier to generate for complex bodies. Further in case of Chimera mesh, one needs to develop a very accurate interpolation schemes in the overlapping regions. Many a times a solver is designed keeping in mind a particular type of grid. Therefore it is not possible in general to use a single solver which can operate on different types of grids. This background provides motivation for the present work and we ask a pertinent and important question whether it is possible to develop a solver which can operate on any type of grid or even on an arbitrary distribution of points. Such an approach has obvious advantages. If one succeeds in developing a grid free solver(i.e. a solver capable of operating on an arbitrary distribution of points) then it is possible to use very simple grid generator for each component of a multicomponent configuration. These can be used to generate distribution of points around a complex configuration. Generation of cloud of points then become much easier, and the problem then is transferred to the solver development. Towards this goal we have selected the Least Squares Kinetic Upwind method, which has the potential to work on any distribution of points. LSKUM can be considered as a generalisation of finite difference method and does not require a regular grid. Ghosh and Deshpande[29, 30, 31] have studied various aspects of LSKUM such as stability, influence of weights on solution accuracy, higher order defect correction technique. Ghosh has demonstarted the ability of the method by computing a variety of 2-D flows and has used only relatively simple meshes for this purpose. We believe that the LSKUM has a much larger potential and to exploit its grid free nature fully, it is neccessary to develop a powerful preprocessor. The primary objective of the present thesis is to

- To exploit fully the potential of 2-D LSKUM by considering a wide variety of point distributions obtained from all types of grids
- To develop a powerful pre-processor for exploiting the grid free nature of LSKUM solver
- To extend the LSKUM to 3-D problems
- Develop and use a new treatment of boundary conditions based on kinetic theory

- Further extend the LSKUM formulation for moving grids and to explore the possibility of developing novel higher order accurate versions of LSKUM

The organisation of the thesis is as follows. In the first chapter we discuss the mathematical formulation. First we give formulation for 2-D LSKUM along with the details for the relevent boundary conditions. This includes a section for the treatment of the farfield boundary condition based on kinetic theory. This is a new addition to the LSKUM and obviously has not been used in the previous work of Ghosh. This is followed by the extension of the basic least squares formulation for 3-D LSKUM. This is an important additional development from the 2-D LSKUM. The subsequent sections give details of the boundary treatmemts for 3-D problem including the new 3-D kinetic treatment of the farfield boundary condition.

In order to exploit fully the grid free nature of LSKUM, we need to develop a powerful preprocessor which enables the solver to operate on any distribution of points. For this purpose we have developed a quad-tree based preprocessor. The full details of the application of this quad-tree structures for the LSKUM preprocessor are explained in the second chapter. A lot of illustrative examples are given to demonstrate the power of the preprocessor. In the third chapter we present results for the test cases based on the use of different types of point distributions obtained from various grid generators.

After demonstrating the grid free nature of 2-D LSKUM we present, in the fourth chapter we present results for the 3-D LSKUM again for a variety of test cases using different types of point distributions. In this chapter we establish the working of 3-D LSKUM for flow past flight vehicles of practicle interest.

In fifth chapter we present further extensions of LSKUM. In this chapter we present two extensions of LSKUM. First is the extension of LSKUM for moving grids. Second, we give a new novel higher order approach for LSKUM which is different from the present two step defect correction method. Results for 1-D and 2-D LSKUM on moving grids are also given in this chapter. Finally we present some preliminary results for 1-D shock tube problem using the novel higher order LSKUM. The concluding remarks are finally given in the last chapter.

1.2 Kinetic Flux Vector Splitting Method

The Kinetic Flux Vector Splitting(KFVS) method [18, 49, 50, 51] is intimatley connected to the Boltzmann equation, which is given by

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} = J \tag{1.1}$$

where f is the velocity distribution function, v is the molecular velocity[1], and x is the molecular position. J represents the collision term which vanishes in the Euler limit.

[1]Here we use the word "molecular" not in the sense of physical molecule we generally understand, but a general term to mean particles or pseudo-molecules

The Euler equations are obtained by taking moments of the Boltzmann equation, when f is a Maxwellian distribution F. The Maxwellian distribution, F, in two dimensions is given by

$$F = \frac{\rho}{I_0}\frac{\beta}{\pi}\exp\left[-\beta(v_1-u_1)^2-\beta(v_2-u_2)^2-I/I_0\right] \tag{1.2}$$

where $\beta = 1/(2RT)$, and I_0 is the internal energy due to non-translational degrees of freedom, $I_0 = \frac{2-\gamma}{\gamma-1}RT$.

We define a moment vector function , Ψ, as

$$\Psi = \left[1, v_1, v_2, I + \frac{v_1^2+v_2^2}{2}\right]^T \tag{1.3}$$

where v_1, v_2 are the Cartesian components of the molecular velocity, I is the internal energy variable, and define the Ψ moment as

$$\langle \Psi, F\rangle \equiv \int_0^\infty dI \int_{-\infty}^\infty dv_1 \int_{-\infty}^\infty dv_2\, \Psi F \tag{1.4}$$

The Euler equations can then be written as

$$\left\langle \Psi, \frac{\partial F}{\partial t} + v_1\frac{\partial F}{\partial x} + v_2\frac{\partial F}{\partial y}\right\rangle = \frac{\partial U}{\partial t} + \frac{\partial}{\partial x}(GX) + \frac{\partial}{\partial y}(GY) = 0 \tag{1.5}$$

where U represents the conserved vector, GX the x-component of the flux vector and GY the y-component of the flux vector. The expressions for these are as given below.

$$\begin{aligned}
U &= [\rho, \rho u_1, \rho u_2, \rho e]^T \\
GX &= [\rho u_1, p+\rho u_1^2, \rho u_1 u_2, u_1(p+\rho e)]^T \\
GY &= [\rho u_2, \rho u_1 u_2, p+\rho u_2^2, u_2(p+\rho e)]^T \\
e &= \frac{RT}{\gamma-1} + \frac{1}{2}\left(u_1^2+u_2^2\right) \\
p &= \rho RT
\end{aligned}$$

Here ρ is the density, u_1, u_2 are the Cartesian components of the fluid velocity, e is the internal energy per unit mass, p is the pressure, T is the temperature, R is the gas constant, and γ is the ratio of the specific heats. The KFVS method involves two levels, the Boltzmann level(theoretical level) and the Euler level(level at which state update operates). The courant splitting is implemented at the Boltzmann level, and then mapped to the Euler level. The courant splitting at the Boltzmann level is achieved by splitting the two components of the molecular velocities v_1 and v_2 into positive and negative parts as

$$v_1 = \frac{v_1+|v_1|}{2} + \frac{v_1-|v_1|}{2}$$

and

$$v_2 = \frac{v_2 + |v_2|}{2} + \frac{v_2 - |v_2|}{2}.$$

The Boltzmann equation thus can be rewritten as

$$\frac{\partial F}{\partial t} + \frac{v_1 + |v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_1 - |v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_2 + |v_2|}{2}\frac{\partial F}{\partial y} + \frac{v_2 - |v_2|}{2}\frac{\partial F}{\partial y} = 0 \tag{1.6}$$

The Ψ moment of the above equation will lead to the Kinetic Flux split Euler equations (called KFVS split Euler equations)

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}(GX^+) + \frac{\partial}{\partial x}(GX^-) + \frac{\partial}{\partial y}(GY^+) + \frac{\partial}{\partial y}(GY^-) = 0 \tag{1.7}$$

where $GX^{\pm}$ and $GY^{\pm}$ represent the split fluxes.

1.3 2-D Least Squares Kinetic Upwind Method

The KFVS split Euler Equations (1.7) forms the starting point for all the KFVS based kinetic schemes. In case of finite volume methods, the above equations are solved for after expressing them in the integral form. Finite difference methods, directly solves the above equations by discretising the various derivatives along the co-ordinate directions. However if we are given an arbitrary distribution of points without any grid structure associated with these points, it will be difficult to discretise the derivatives as is normally done in the finite difference approaches. It is in this context we have used the least squares approach of Ghosh[31], for the estimation of derivatives for any arbitrary distribution of points. The gradients terms $\frac{\partial F}{\partial x}$ and $\frac{\partial F}{\partial y}$ which appear in Boltzmann equation (1.6), are evaluated using the values at the surrounding points as shown in fig. 1.1. We give here the analysis due to Ghosh & Deshpande [30]. If there is a point o surrounded by n points, then Taylor series around "o" gives for any quantity F gives us,

$$F_i = F_o + F_{xo}\,\Delta x_i + F_{y_o}\,\Delta y_i + \text{h.o.t}, \qquad i = 1, \cdots, n, \tag{1.8}$$

where $\Delta x_i = x_i - x_o$ and $\Delta y_i = y_i - y_o$ Minimizing the square of the error E, defined by

$$E \equiv \sum_{i=1}^{n} \left(\Delta F_i - F_{xo}\Delta x_i - F_{y_o}\Delta y_i\right)^2, \tag{1.9}$$

where $\Delta F_i = F_i - F_o$, gives the following formulae for the gradients

$$\begin{aligned} F_{xo} &= \frac{\sum \Delta y_i^2 \sum \Delta x_i \Delta F_i - \sum \Delta x_i \Delta y_i \sum \Delta y_i \Delta F_i}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i} \\ F_{y_o} &= \frac{\sum \Delta x_i^2 \sum \Delta y_i \Delta F_i - \sum \Delta x_i \Delta y_i \sum \Delta x_i \Delta F_i}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i}, \end{aligned} \tag{1.10}$$

where, $\sum$ represents the summation over all the points in the neighbourhood.

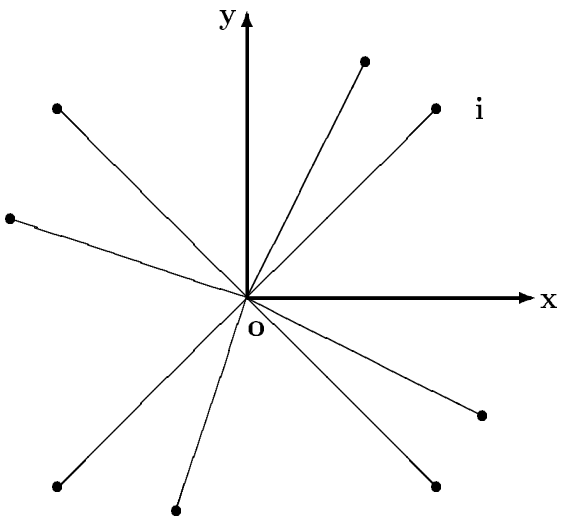


Figure 1.1: Sketch of typical point connectivity for LSKUM

The formulae given in equation (1.10) can now be used to obtain the point values of F_{x_o} and F_{y_o} throughout the field.

The exact solution to the 2-D Boltmann equation

$$\frac{\partial F}{\partial t} + v_1 \frac{\partial F}{\partial x} + v_2 \frac{\partial F}{\partial y} = 0$$

is given by

$$F(t + \Delta t, x, y, v_1, v_2) = F(t, x - v_1 \Delta t, y - v_2 \Delta t, v_1, v_2) \tag{1.11}$$

This indicates that the propagation of information from any node P_i to the recieving node P_o depends upon the location of the node P_i relative to P_o and the signs of v_1 and v_2. If $v_1 > 0$ then only nodes lying to the left of P_o will influence the solution at P_o. Similarly for $v_1 < 0$ only nodes to the right of P_o will influence the solution at P_o . Using similar arguments, we can show that for $v_2 > 0$, the solution at P_o is influenced by only nodes lying below it, while for $v_2 < 0$ it is influenced by nodes above P_o. This property called the signal propagation property should be taken into account while developing an upwind scheme.

Consider the term $\frac{v_1+|v_1|}{2}\frac{\partial F}{\partial x}$ which appears in the equation (1.6). The molecular velocity multiplying the derivative is always positive for $v_1 > 0$, while it is zero for $v_1 < 0$. This indicates that the least squares approximation for the derivative $\frac{\partial F}{\partial x}$ at node P_o for $v_1 > 0$ should use data of F at nodes lying only to the left of P_o.

This ensures that the signal propagation property is not violated. All such nodes are referred to as the connectivity substencil indicated as $N_1(P_o)$. Various substencils can be seen in fig. 1.2. The derivative evaluated so is identified as $\frac{\partial F}{\partial x} = (\frac{\partial F}{\partial x})_{N_1(P_o)}$. By similar arguments we can easily see that for $v_1 < 0$ $\frac{\partial F}{\partial x} = (\frac{\partial F}{\partial x})_{N_2(P_o)}$. For the y-derivatives we have, for $v_2 > 0$ $\frac{\partial F}{\partial y} = (\frac{\partial F}{\partial x})_{N_3(P_o)}$ and for $v_2 < 0$ $\frac{\partial F}{\partial y} = (\frac{\partial F}{\partial x})_{N_4(P_o)}$. The split Boltzmann equation (1.6) can thus be represented as

$$\frac{\partial F}{\partial t} + \frac{v_1 + |v_1|}{2}\left(\frac{\partial F}{\partial x}\right)_{N_1(P_o)} + \frac{v_1 - |v_1|}{2}\left(\frac{\partial F}{\partial x}\right)_{N_2(P_o)} + \frac{v_2 + |v_2|}{2}\left(\frac{\partial F}{\partial y}\right)_{N_3(P_o)} + \frac{v_2 - |v_2|}{2}\left(\frac{\partial F}{\partial y}\right)_{N_4(P_o)} = 0 \tag{1.12}$$

We can now use the above analysis to develop the first order LSKUM. Discretising the time derivative to first order and taking ψ moment of the above equation we get

$$U^{n+1} = U^n - \Delta t \left[\left(\frac{\partial}{\partial x}[GX^+]\right)^n_{N_1(P_o)} + \left(\frac{\partial}{\partial x}[GX^-]\right)^n_{N_2(P_o)} + \left(\frac{\partial}{\partial y}[GY^+]\right)^n_{N_3(P_o)} + \left(\frac{\partial}{\partial y}[GY^-]\right)^n_{N_4(P_o)}\right] \tag{1.13}$$

where

$$\left(\frac{\partial}{\partial x}[GX^+]\right)_{N_1(P_o)} = \left\{\frac{\sum \Delta y_i^2 \sum \Delta x_i \Delta GX_i^+ - \sum \Delta x_i \Delta y_i \sum \Delta y_i \Delta GX_i^+}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i}\right\}_{N_1(P_o)}$$

$$\left(\frac{\partial}{\partial x}[GX^-]\right)_{N_2(P_o)} = \left\{\frac{\sum \Delta y_i^2 \sum \Delta x_i \Delta GX_i^- - \sum \Delta x_i \Delta y_i \sum \Delta y_i \Delta GX_i^-}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i}\right\}_{N_2(P_o)}$$

$$\left(\frac{\partial}{\partial y}[GY^+]\right)_{N_3(P_o)} = \left\{\frac{\sum \Delta x_i^2 \sum \Delta y_i \Delta GY_i^+ - \sum \Delta x_i \Delta y_i \sum \Delta x_i \Delta GY_i^+}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i}\right\}_{N_3(P_o)}$$

$$\left(\frac{\partial}{\partial y}[GY^-]\right)_{N_4(P_o)} = \left\{\frac{\sum \Delta x_i^2 \sum \Delta y_i \Delta GY_i^- - \sum \Delta x_i \Delta y_i \sum \Delta x_i \Delta GY_i^-}{\sum \Delta x_i^2 \sum \Delta y_i^2 - \sum \Delta x_i \Delta y_i}\right\}_{N_4(P_o)}$$

$\Delta GX_i^{\pm} = GX_i^{\pm} - GX_o^{\pm}$ and $\Delta GY_i^{\pm} = GY_i^{\pm} - GY_o^{\pm}$

Δt is the time step and $GX^{\pm}$ and $GY^{\pm}$ represent the split fluxes. The expressions for these are given in appendix A.

1.3.1 2-D Solid wall Boundary condition

Consider a boundary surface as shown in fig. 1.3. Here x, y represents the local co-ordinate system at point P, where x, is along the tangent and y along the normal. For the treatment of surface tangency boundary condition on solid surfaces, the velocity distribution function f at P is constructed as the union of two half Maxwellians F_I and F_R,

$$f_P = F_I \cup F_R,$$

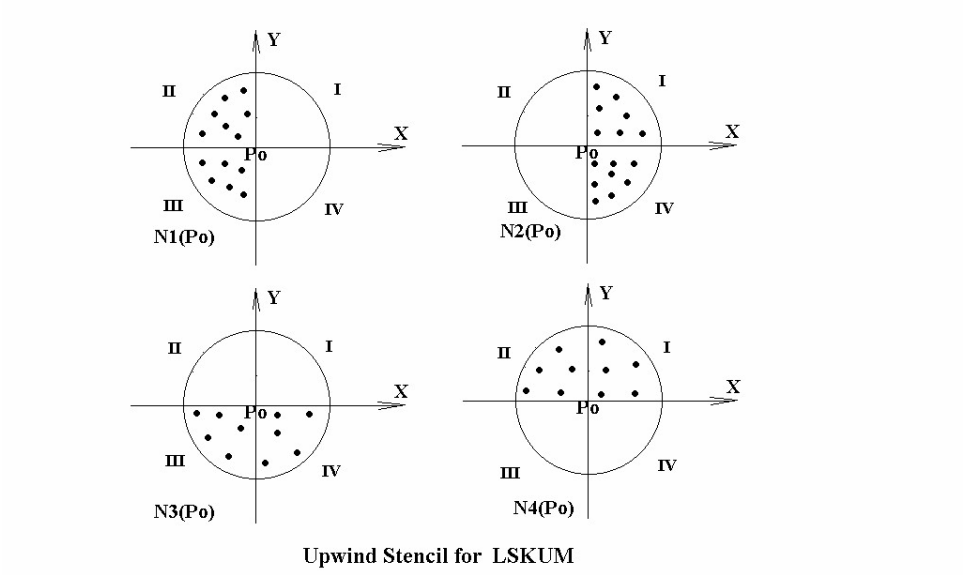


Figure 1.2: Stencil splitting for LSKUM

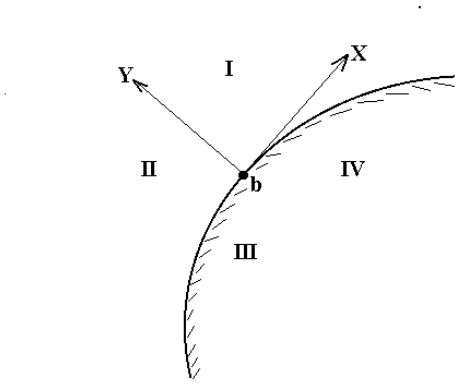


Figure 1.3: A typical 2-D boundary

where F_I and F_R corresponds to the incident and reflected molecules respectively. Following the principle of specular reflection [49, 50] of kinetic theory of gases, F_R is reconstructed from F_I by reversing the component of velocity in the direction normal to the surface. Let F be the Maxwellian given by

$$F = \frac{\rho}{I_0}(\frac{\beta}{\pi}) \exp\left[-\beta(v_1 - u_1)^2 - \beta(v_2 - u_2)^2 - I/I_0\right],$$

where v_1, v_2 are the velocity components of the molecule and u_1, u_2 are the components of fluid velocity along x, y. F_I and F_R are now expressed as

$$\begin{aligned} F_I &= F_I(v_1, v_2) \quad for \;\; v_2 < 0 \\ F_R &= F_I(v_1, -v_2) \quad for \;\; v_2 > 0. \end{aligned}$$

The time update for f_P is given by

$$f_P^{n+1} = F_I^{n+1} \cup F_R^{n+1}. \tag{1.14}$$

The F_I^{n+1} and F_R^{n+1} are expressed as

$$\begin{aligned} F_I^{n+1} &= F^n(v_2) \quad - \quad \Delta t\,(v_1 F_{x_1} + v_2 F_{x_2}) \quad for \;\; v_2 < 0 \\ F_R^{n+1} &= F^n(-v_2) - \Delta t\,(v_1 F_{x_1} - v_2 F_{x_2}) \quad for \;\; v_2 > 0 \end{aligned} \tag{1.15}$$

Substituting equation (1.15) in (1.14) and taking Ψ moments of the resulting equation, we get the update scheme for points on solid boundaries as

$$\begin{aligned} U_P^{n+1}(l) &= U_P^n(l) - 2\;\Delta t\left(\frac{\partial GX^{II}(l)}{\partial x} + \frac{\partial GX^{I}(l)}{\partial x} + \frac{\partial GY^{II}(l)}{\partial y} + \frac{\partial GY^{I}(l)}{\partial y}\right) \text{ for } l = 1, 2, \&4 \\ &= 0 \quad \text{for } \; l = 3 \end{aligned}$$

where Δt is the time step, GX^I,GX^{II} and GY^I,GY^{II} represents the x and y quadrant split fluxes repectively. For example GX^I represents x quadrant split fluxes in the first quadrant, i.e. for $v_1 < 0$, $v_2 < 0$. Similarly GX^{II},GX^{III} and GX^{IV} represents x quadrant split fluxes in the second quadrant($v_1 > 0$, $v_2 < 0$), third($v_1 > 0$, $v_2 > 0$) and fourth quadrant($v_1 < 0$, $v_2 > 0$) respectively. The expressions for these various quadrant split fluxes are given in appendix A

1.3.2 2-D Far Field Boundary condition

The treatment of the far field boundary condition[66] is very similar to previous one. Referring to fig. 1.3, the velocity distribution function f_P at any point P lying on the outer boundary surface is constructed by the union of two half Maxwellians F_{in} and F_{out}. F_{in} corresponds to the incoming particles having negative velocity with respect to the direction normal to the boundary. The velocity distribution function for

this part corresponds to the freestream conditions. F_{out} corresponds to the outgoing particles having positive velocity in the normal direction. For the outgoing particles the velocity distribution function is constructed from the interior region. Thus we can write the time update for f_P as

$$f_P^{n+1} = F_{in}^{n+1} \cup F_{out}^{n+1}. \tag{1.16}$$

F_{in}^{n+1} and F_{out}^{n+1} can be expressed as,

$$\begin{aligned} F_{in}^{n+1} &= F(v_1, v_2) \quad for \quad v_2 < 0 \\ &= F_\infty \quad for \quad all \quad n \\ &= \frac{\rho_\infty}{I_{0_\infty}}(\frac{\beta_\infty}{\pi}) \exp\left[-\beta(v_1 - u_{1_\infty})^2 - \beta(v_2 - u_{2_\infty})^2 - I/I_{0_\infty}\right] \\ F_{out}^{n+1} &= F^n - \Delta t\,(v_1 F_{x_1} + v_2 F_{x_2}) \quad for \quad v_2 > 0 \end{aligned} \tag{1.17}$$

substituting equation(1.17) in (1.16) and taking Ψ moments of the resulting equation we get the update scheme for points lying on the outer boundary as

$$U_P^{n+1} = \overline{U}^n - \quad \Delta t \left(\frac{\partial GX^{III}}{\partial x} + \frac{\partial GX^{IV}}{\partial x} + \frac{\partial GY^{III}}{\partial y} + \frac{\partial GY^{IV}}{\partial y} \right) \tag{1.18}$$

where Δt is the time step, GX^{III},GX^{IV} and GY^{III},GY^{IV} represent the quadrant split fluxes in the third and fourth quadrants respectively. The expressions for these fluxes and the term $\overline{U}^n$(which is a function of freestream and interior values) can be seen in Appendix A

1.4 3-D Least Squares Kinetic Upwind Method

Least squares approach for 3-D is very much similar to the 2-D formulation described in the previous section. Consider the Boltzmann equation

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} = J \tag{1.19}$$

where f is the velocity distribution function, v is the molecular velocity and x is the molecular position. J represents the collision term which vanishes in the Euler limit. The Euler equations are obtained by taking moments of the Boltzmann equation, when f is a Maxwellian distribution. The Maxwellian distribution, F, in three dimensions is given by

$$F = \frac{\rho}{I_0}(\frac{\beta}{\pi})^{\frac{3}{2}} \exp\left[-\beta(v_1 - u_1)^2 - \beta(v_2 - u_2)^2 - \beta(v_3 - u_3)^2 - I/I_0\right] \tag{1.20}$$

where v_1, v_2, v_3 are the cartesian components of the molecular velocity, u_1, u_2, u_3 are the cartesian components of the fluid velocity, $\beta = 1/(2RT)$, and I_0 is the internal energy due to non-translational degrees of freedom, $I_0 = \frac{5-3\gamma}{2(\gamma-1)}RT$. The three

dimensional Euler equations can be written as

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}[GX] + \frac{\partial}{\partial y}[GY] + \frac{\partial}{\partial z}[GZ] = 0 \tag{1.21}$$

where

$$\begin{aligned} U &= [\rho, \rho u_1, \rho u_2, \rho u_3, \rho e]^T \\ GX &= [\rho u_1, p + \rho u_1^2, \rho u_1 u_2, \rho u_1 u_3, u_1(p + \rho e)]^T \\ GY &= [\rho u_2, \rho u_1 u_2, p + \rho u_2^2, \rho u_2 u_3, u_2(p + \rho e)]^T \\ GZ &= [\rho u_3, \rho u_1 u_3, \rho u_2 u_3, p + \rho u_3^2, u_3(p + \rho e)]^T \\ e &= \frac{RT}{\gamma - 1} + \frac{1}{2}\left(u_1^2 + u_2^2 + u_3^2\right) \\ p &= \rho RT \end{aligned}$$

Here ρ is the density, u_1, u_2, u_3 are the Cartesian components of the fluid velocity, e is the internal energy per unit mass, p is the pressure, T is the temperature, R is the gas constant, and γ is the ratio of the specific heats.

As before we define a moment function vector, Ψ, as

$$\Psi = \left[1, v_1, v_2, v_3, I + \frac{v_1^2 + v_2^2 + v_3^2}{2}\right]^T \tag{1.22}$$

where v_1, v_2, v_3 are the Cartesian components of the molecular velocity, I is the internal energy variable, and define the moment

$$\langle \Psi, F \rangle \equiv \int_0^\infty dI \int_{-\infty}^\infty dv_1 \int_{-\infty}^\infty dv_2 \int_{-\infty}^\infty dv_3 \, \Psi F \tag{1.23}$$

The Euler equations are then obtained as

$$\left\langle \Psi, \frac{\partial F}{\partial t} + v_1 \frac{\partial F}{\partial x} + v_2 \frac{\partial F}{\partial y} + v_3 \frac{\partial F}{\partial z} \right\rangle = \frac{\partial U}{\partial t} + \frac{\partial}{\partial x}[GX] + \frac{\partial}{\partial y}[GY] + \frac{\partial}{\partial z}[GZ] = 0 \tag{1.24}$$

The Courant splitting is implemented at the Boltzmann level, and then mapped to the Euler level. The Courant splitting at the Boltzmann level is achieved by splitting the three components of the molecular velocities v_1, v_2 and v_3 into positive and negative parts as

$$v_1 = \frac{v_1 + |v_1|}{2} + \frac{v_1 - |v_1|}{2},$$

$$v_2 = \frac{v_2 + |v_2|}{2} + \frac{v_2 - |v_2|}{2}$$

and

$$v_3 = \frac{v_3 + |v_3|}{3} + \frac{v_3 - |v_3|}{2}.$$

In the Euler limit, the Boltzmann equation thus can be rewritten as

$$\frac{\partial F}{\partial t} + \frac{v_1 + |v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_1 - |v_1|}{2}\frac{\partial F}{\partial x} + \frac{v_2 + |v_2|}{2}\frac{\partial F}{\partial y} + \frac{v_2 - |v_2|}{2}\frac{\partial F}{\partial y} + \frac{v_3 + |v_3|}{2}\frac{\partial F}{\partial z} + \frac{v_3 - |v_3|}{2}\frac{\partial F}{\partial z} = 0 \qquad (1.25)$$

The Ψ moments of the above equation will lead to the Euler equations in the split flux form as

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}[GX^+] + \frac{\partial}{\partial x}[GX^-] + \frac{\partial}{\partial y}[GY^+] + \frac{\partial}{\partial y}[GY^-] + \frac{\partial}{\partial z}[GZ^+] + \frac{\partial}{\partial z}[GZ^-] = 0 \quad (1.26)$$

where $GX^{\pm}$, $GY^{\pm}$ and $GZ^{\pm}$ represent the split fluxes. These split fluxes are functions of U. The expressions for these are given in Appendix B. The above split Euler equations are solved using the least squares approach which is very much similar to the 2-D approach explained previously. The derivatives of the various split fluxes are evaluated using the least squares approximation. The time derivative is descritized using a suitable time marching scheme. This finally leads us to the 3-D LSKUM scheme for Euler equations. In the next para we explain the details of the least squares approximation of derivatives for 3-D case.

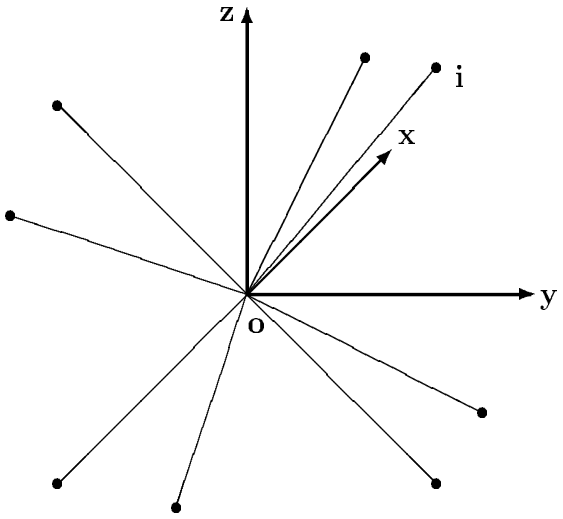


Figure 1.4: Sketch of typical point connectivity for 3D-LSKUM

Let F be any quantity of interest which is available at all the nodes. We are interested in finding the derivatives of F at all the nodes. In order to obtain them we proceed as follows. Consider a point o surrounded by n points as shown in fig. 1.4, then Taylor series around o for F at any point i in the neighbourhood of o gives us,

$$F_i = F_o + F_{xo}\,\Delta x_i + F_{y_o}\,\Delta y_i + F_{zo}\,\Delta z_i + \text{h.}o.t, \qquad i = 1, \cdots, n, \qquad (1.27)$$

where $\Delta x_i = x_i - x_o$, $\Delta y_i = y_i - y_o$ and $\Delta z_i = z_i - z_o$.

When the number of points are more than three for the above case this represents a classical overdetermined system for the unknowns $F_{x_o}, F_{y_o}, F_{z_o}$. Now in order to determine them we minimize the square of the error E w.r.t to these unknowns. The error E is defined as

$$E \equiv \sum_{i=1}^{n} \left(\Delta F_i - F_{x_o} \Delta x_i - F_{y_o} \Delta y_i - F_{z_o} \Delta z_i \right)^2, \tag{1.28}$$

where $\Delta F_i = F_i - F_o$.

The following formulae are then obtained for the gradients

$$\frac{F_{x_o}}{D1} = \frac{F_{y_o}}{D2} = \frac{F_{z_o}}{D3} = \frac{1}{D} \tag{1.29}$$

where

$$D1 = - \begin{vmatrix} \sum \Delta x_i \Delta y_i & \sum \Delta z_i \Delta x_i & \sum \Delta x_i \Delta F_i \\ \sum \Delta y_i^2 & \sum \Delta z_i \Delta y_i & \sum \Delta y_i \Delta F_i \\ \sum \Delta y_i \Delta z_i & \sum \Delta z_i{}^2 & \sum \Delta z_i \Delta F_i \end{vmatrix}$$

$$D2 = \begin{vmatrix} \sum \Delta x_i^2 & \sum \Delta z_i \Delta x_i & \sum \Delta x_i \Delta F_i \\ \sum \Delta x_i \Delta y_i & \sum \Delta z_i \Delta y_i & \sum \Delta y_i \Delta F_i \\ \sum \Delta x_i \Delta z_i & \sum \Delta z_i^2 & \sum \Delta z_i \Delta F_i \end{vmatrix}$$

$$D3 = - \begin{vmatrix} \sum \Delta x_i^2 & \sum \Delta x_i \Delta y_i & \sum \Delta x_i \Delta F_i \\ \sum \Delta x_i \Delta y_i & \sum \Delta y_i^2 & \sum \Delta y_i \Delta F_i \\ \sum \Delta x_i \Delta zi & \sum \Delta y_i \Delta z_i & \sum \Delta z_i \Delta F_i \end{vmatrix}$$

$$D = - \begin{vmatrix} \sum \Delta x_i^2 & \sum \Delta x_i \Delta y_i & \sum \Delta z_i \Delta yi \\ \sum \Delta x_i \Delta y_i & \sum \Delta y_i^2 & \sum \Delta x_i \Delta y_i \\ \sum \Delta x_i \Delta x_i & \sum \Delta y_i \Delta z_i & \sum \Delta z_i^2 \end{vmatrix}$$

The above formulae gives us first order accurate F_x^1, F_y^1 and F_z^1 at point o. In order to evaluate these for second order accuracy, we follow the two step approach of Ghosh and Deshpande [31]. First step consists of using first order formula 1.29 to evaluate the derivatives. The second step involves the defect correction step where the quantity ΔF_i is replaced by $\widetilde{\Delta F_i}$ and the first order formulae 1.29 repeated once more with the modified differences $\widetilde{\Delta F_i}$ given by

$$\widetilde{\Delta F_i} = \Delta F_i - \frac{1}{2} \left(\Delta F_{x_i}^1 \, \Delta x_i + \Delta F_{y_i}^1 \, \Delta y_i + \Delta F_{z_i}^1 \, \Delta z_i \right)$$

where $\Delta F^1_{x\,i} = F^1_{x\,i} - F^1_{x\,o}$, $\Delta F^1_{y\,i} = F^1_{y\,i} - F^1_{y\,o}$ and $\Delta F^1_{z\,i} = F^1_{z\,i} - F^1_{z\,o}$

The derivative at any point is a local property, hence in order to give more weightage to points closer, in the calculation of these, a weighted least squares approach is used. Here all the summation terms such as $\sum \Delta x_i \Delta y_i, \sum \Delta z_i \Delta x_i, \sum \Delta x_i \Delta F_i \cdots$ etc which appear in the formulae are replaced by $\sum W_i \Delta x_i \Delta y_i$, where W_i is a function of the distance d_i, between any point i and point o. The weight function[31] considered in the present case is $W_i = \frac{1}{d_i^p}$. It has been shown by Ghosh[31] that the optimum value of p is 6, and this is the value we have used in our present computations.

Using the above procedure we determine the various derivatives of the Maxwellian F appearing in equation (1.25). Thus we can write the update scheme for F as

$$F^{n+1} = F^n - \Delta t \left(\frac{v_1 + |v_1|}{2} \frac{\partial F}{\partial x}^n + \frac{v_1 - |v_1|}{2} \frac{\partial F}{\partial x}^n + \frac{v_2 + |v_2|}{2} \frac{\partial F}{\partial y}^n + \frac{v_2 - |v_2|}{2} \frac{\partial F}{\partial y}^n + \frac{v_3 + |v_3|}{2} \frac{\partial F}{\partial z}^n + \frac{v_3 - |v_3|}{2} \frac{\partial F}{\partial z}^n \right) \quad (1.30)$$

where Δt is the time step, n represents previous time level and $n+1$ the present time level.

The Ψ moments of the above equation gives us the state update formula as

$$U^{n+1} = U^n - \Delta t \left(\frac{\partial}{\partial x}[GX^+]^n + \frac{\partial}{\partial x}[GX^-]^n + \frac{\partial}{\partial y}[GY^+]^n + \frac{\partial}{\partial y}[GY^-]^n + \frac{\partial}{\partial z}[GZ^+]^n + \frac{\partial}{\partial z}[GZ^-]^n \right) \quad (1.31)$$

1.5 3-D Treatment of Boundary Conditions

In this section we present the details of the surface tangency boundary condition at the solid wall and the far field boundary condition. In the present study both these boundary conditions are treated at the molecular level itself. The surface tangency condition uses the specular reflection principle. For the far field boundary condition we have developed a new approach which is also at the kinetic level[67]. Typically far field boundary condition requires different treatment depending on the flow being subsonic inflow, subsonic outflow, supersonic inflow or supersonic outflow. The present far field boundary condition does not depend on the these conditions at the far field boundary and the flux expressions remain identically the same for all cases.

1.5.1 3-D Solid wall Boundary condition

Consider a three dimensional surface as shown in fig. 1.5. x, y, z represents the local co-ordinate system at point P, where z is the outward normal to the surface at P and x,y are two mutually perpendicular tangent vectors in the plane tangent to the surface at point P. For the treatment of surface tangency boundary condition on

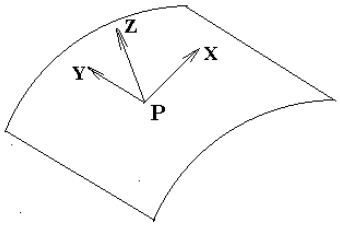


Figure 1.5: A typical 3-D boundary

solid surfaces, the velocity distribution function f_P at any point P is constructed as the union of two half Maxwellians F_I and F_R,

$$f_P = F_I \cup F_R,$$

where F_I and F_R corresponds to the incident and reflected molecules respectively. Following the principle of specular reflection [49, 50] of kinetic theory, F_R is reconstructed from F_I by reversing the component of velocity in the direction normal to the surface. If F is the Maxwellian given by

$$F = \frac{\rho}{I_0}(\frac{\beta}{\pi})^{\frac{3}{2}} \exp\left[-\beta(v_1-u_1)^2 - \beta(v_2-u_2)^2 - \beta(v_3-u_3)^2 - I/I_0\right],$$

where v_1, v_2, v_3 are the velocity componenets of the molecule and u_1, u_2, u_3 are the components of fluid velocity along x, y, z. then F_I and F_R can be expressed as

$$\begin{aligned} F_I &= F_I(v_1, v_2, v_3) \quad for \;\; v_3 < 0 \\ F_R &= F_I(v_1, v_2, -v_3) \quad for \;\; v_3 > 0. \end{aligned}$$

Thus the time update for f_P is written as

$$f_P^{n+1} = F_I^{n+1} \cup F_R^{n+1}. \tag{1.32}$$

Now F_I^{n+1} and F_R^{n+1} are written as

$$\begin{aligned} F_I^{n+1} &= F^n(v_3) - \Delta t\,(v_1F_x + v_2F_y + v_3F_z) \quad for \;\; v_3 < 0 \\ F_R^{n+1} &= F^n(-v_3) - \Delta t\,(v_1F_x + v_2F_y - v_3F_z) \quad for \;\; v_3 > 0 \end{aligned} \tag{1.33}$$

substituting equation (1.33) in (1.32) and taking Ψ moments of the resulting equation, we get the update scheme for points on solid boundaries as

$$\begin{aligned} U_P^{n+1}(l) &= U_P^n(l) - 2\ \Delta t \left(\frac{\partial}{\partial x}[GX^{+\cdot\,-}(l)] + \frac{\partial}{\partial x}[GX^{-\cdot\,-}(l)] \right. \\ &+ \left. \frac{\partial}{\partial y}[GY^{\cdot\,+\,-}(l)] + \frac{\partial}{\partial y}[GY^{\cdot\,-\,-}(l)] + \frac{\partial}{\partial z}[GZ^{-}(l)\Big) \right] \text{ for } l = 1, 2, 3\&5 \\ &= 0 \quad \text{for } l = 4 \end{aligned} \tag{1.34}$$

where Δt is the time step, the superscripts $+\&-$ indicates the sign of the split part, The dot symbol, $\cdot$ indicates the full range and the position of these indicates the component number. For example $\frac{\partial GX^{+\cdot\,-}}{\partial X}$ represents GX quadrant split fluxes for $v_1 > 0$, $-\infty < v_2 < \infty$ and $v_3 < 0$. The expressions for various quadrant split fluxes are given in Appendix B.

1.5.2 3-D Far Field Boundary condition

The treatment of the far field boundary condition is very similar to previous one. Consider fig. 1.5, the velocity distribution function f_P at any point P lying on the outer boundary surface is constructed as the union of two half Maxwellians corresponding to the incoming and the outgoing particles. The incoming particles have negative velocity with respect to the direction normal to the boundary. The velocity distribution function F_{in} for this part corresponds to the freestream conditions. The outgoing particles have positive velocity in the normal direction. For the outgoing particles the velocity distribution function F_{out} is constructed from the interior region. Thus we can write the time update for f_P as

$$f_P^{n+1} = F_{in}^{n+1} \cup F_{out}^{n+1}. \tag{1.35}$$

F_{in}^{n+1} and F_{out}^{n+1} can be expressed as,

$$\begin{aligned} F_{in}^{n+1} &= F(v_1, v_2, v_3) \ \ for \ \ v_3 < 0 \\ &= F_\infty \ \ for \ \ all \ \ n \\ &= \frac{\rho_\infty}{I_{0_\infty}} \left(\frac{\beta_\infty}{\pi}\right)^{\frac{3}{2}} \exp\left[-\beta(v_1 - u_{1_\infty})^2 - \beta(v_2 - u_{2_\infty})^2 - \beta(v_3 - u_{3_\infty})^2 - I/I_{0_\infty}\right] \\ F_{out}^{n+1} &= F^n - \Delta t \left(v_1 F_{x_1} + v_2 F_{x_2} + v_3 F_{x_3}\right) \quad for \ \ v_3 > 0 \end{aligned} \tag{1.36}$$

substituting equation(1.36) in (1.35) and taking Ψ moments of the resulting equation we get the update scheme for points lying on the outer boundary as

$$\begin{aligned} U_P^{n+1} &= \overline{U}_P^n - \ \Delta t \left(\frac{\partial}{\partial x}[GX^{+\cdot\,+}] + \frac{\partial}{\partial x}[GX^{-\cdot\,+}] \right. \\ &+ \left. \frac{\partial}{\partial y}[GY^{\cdot\,+\,+}] + \frac{\partial}{\partial y}[GY^{\cdot\,-\,+}] + \frac{\partial}{\partial z}[GZ^{+}]\right) \end{aligned} \tag{1.37}$$

The expressions for $\overline{U}_P^n$ and various quadrant split fluxes are given in Appendix B.

We can easily observe from section1.3.2 and section1.5.2, that the far field boundary treatment using KOBC is very similar in both 2D and 3D formulations. Also the various flux formulae for the far field boundary treatment(given in appendix A and appendix B) are also very much similar. A very important observation we should make here is that, these formulae essentially remain the same for all subsonic inflow/outflow supersonic inflow/outflow conditions at the outer boundary. In this sense KOBC is a more generalised far field boundary condition independent of the flow conditions. While this is not true in the case of Riemann type of far field boundary conditions, where different boundary conditions will have to be applied based upon the flow conditions such as subsonic inflow/outflow supersonic inflow/outflow. Also in the conventional Riemann type approach the boundary conditions are applied based on the flow conditions assumed to be one dimensional in the direction normal to the outer boundary. But in the case of KOBC we can very cleary see, it makes no such assumptions for both 2D as well as 3D farfield boundary treatment. For these reasons we feel, the present KOBC is a more generalised boundary treatment for far field boundary condition.

1.6 Stability criterion

Consider 1-D Boltzmann equation given by

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} = J \tag{1.38}$$

Following the stability analysis of Ghosh[31], we get the stability condition as

$$\left| \frac{v\Delta t}{\Delta x} \right| \leq 1 \tag{1.39}$$

for a first order upwind scheme for the above equation (1.38). This condition apparently imposes a severe restriction on v for finite values of Δt and Δx ; ie

$$-\frac{\Delta x}{\Delta t} \leq \quad v \quad \leq \frac{\Delta x}{\Delta t} \tag{1.40}$$

The moments in the moment method strategy (which is the basis for converting different parameters from the Boltzmann level to the Euler level) in the KFVS formulation involve a velocity range from $-\infty$ to $+\infty$. Thus, the moments do not recieve contributions due to velocities outside the cut-off bound (1.40). However, the contributions from these velocities to the moments become negligibly small because of the rapid exponential decay of F. Note that the velocity distribution function F is a normal distribution in v with mean u and standard deviation $\sigma(=\sqrt{RT})$. Hence, an integration of F(first moment) over a 3σ limit for v covers 99.73% of the area under the curve, which is a very good approximation to the moments of F over $-\infty$ to $+\infty$

limit. Also further Mandal[51] has found out that the second, third, fourth and fifth moments with a 3σ limit covers, respectively 99.73%, 97.92%, 97.43% and 93.2% of the total area. Therefore, the maximum value of v that needs to be considered is $u + 3\sigma$, to obtain an equivalent stability condition, ie

$$\left| \frac{(u + 3\sigma)\Delta t}{\Delta x} \right| \leq 1 \tag{1.41}$$

which reduce to the condition

$$\Delta t \leq \frac{\Delta x}{|u| + 3\sigma}. \tag{1.42}$$

For a second order LSKUM scheme, Ghosh[31] has found out that the stability condition is very much similar to the equation (1.42) and can be written as

$$\Delta t \leq \delta \frac{\Delta x}{|u| + 3\sigma}, \tag{1.43}$$

where $\delta = \frac{1}{10}$. We can thus use the stability condition as

$$\Delta t \leq \text{rlax} \frac{\Delta x}{|u| + 3\sigma}, \tag{1.44}$$

where the relaxation factor rlax would take into account the δ factor for second order scheme.

The above stability critirion for 1-D LSKUM is the extended heuristically to multidimensional case. As an example, consider the 3-D case as shown in fig. 1.4. each ray connecting point o with its neighbouring point i is locally considered as a 1-D case. Then the stability limit Δt_i for each ray is obtained using relation (1.44). Final stability criterion is then simply given by

$$\Delta t = \text{Min of all i} \frac{\Delta s_i}{|q_o| + 3\sigma} \tag{1.45}$$

where Δs_i is the distance between point o and point i, q_o is the total velocity at point o.

Chapter 2

Quad-tree Pre-processor

2.1 Introduction

From the discussions in the previous chapters it is evident that the central problem in LSKUM is to determine discrete approximations to derivatives $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$ at any node P_o using data at the neighbouring points. The distribution of points around P_o can be completely arbitrary. We define the set of neighbours (also called stencil) of P_o by

$$N(P_o) = \{ \text{ all } P_i \text{ such that } d(P_o, P_i) < h\}$$

where $d(P_o, P_i)$ = Eucledian distance between P_o and P_i and h= a characteristic linear dimension of the neighbourhood $N(P_o)$ to be specified by the user. Exact definition of the neighbourhood is not important to LSKUM as long a P_i are local to P_o. The discrete approximations to the derivatives $\left(\frac{\partial f}{\partial x}\right)_o, \left(\frac{\partial f}{\partial y}\right)_o$ are determined by using least squares approximation. These in turn are used in the update formulae to obtain the solution at the next time level. Given any arbitrary distribution of nodes(which we may obtain form any type of grid generator or by some other method), we need the data of the immediate surrounding nodes defined as the connectivity data or simply the connectivity of the nodes. In this chapter we address this crucial issue of obtaining the connectivity data in an efficient way using standard quad tree data structures.

2.2 Connectivity and its importance for LSKUM solver

Generation of connectivity data can be a tricky job. This task is usually done using a pre-processor. In case of a structured grid, the pre-processor can easily generate the connectivity data by selecting the points in the immediate neighbourhood along the two co-ordinate directions as shown in fig. 2.1. However this procedure would be specific to structured grids as the pre-processor makes use of the regularity of the grid. If we are considering the nodes generated by an Unstructured grid, then the pre-processor will have to adopt a different logic to get the connectivity data. This will

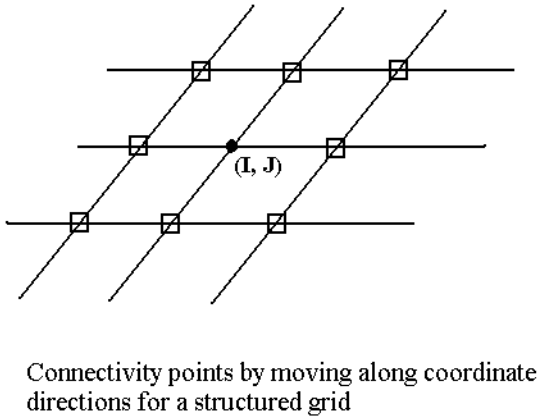
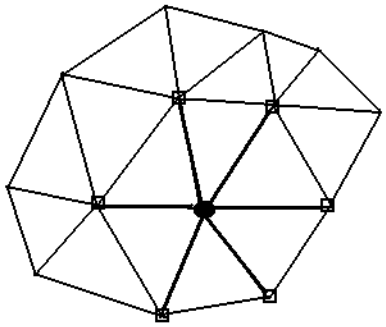


Figure 2.1: Connectivity definition for nodes from structured grid

now be entirely different from the method adopted for points of structured grid. In the case of points generated from unstructured grids, the pre-processor can possibly use the Edge based data structure[80] for the connectivity generation as shown in fig. 2.2. Such pre-processors make use of properties specific to the grid from which the nodes were obtained. Now consider a distribution of points obtained by using a combination of different types of grids, writing a pre-processor would then become more complicated if we were to take into account the properties of each of the grid used. Therefore it would be desirable to have a pre-processor which would operate on any type of distribution of points independent of the type of grid from which these are generated. This motivation led us to the developemt of a pre-processor based on quad tree data structure.

There are certain minimum attributes that the connectivity data should posses. They are, availability of a certain minimum number of points for least squares approach to be meaningful and effective blocking of the certain regions from the neighbourhood which we refer to as eclipsed region. It must be emphasised here that these eclipsed regions can cause considerable havoc in the solution update. Before we go into the details of all these aspects, let us first consider the issue of blocking the eclipsed region.

Consider a point P as shown in fig. (2.3). This particular point lies on the lower side of the airfoil near the trailing edge as shown in the figure. The neighbourhood for this point is defined by the rectangle as shown in this figure. It can be easily observed from the figure, that all the points contained in this rectangle cannot be considered as the connectivity data for the point P. This is because points lying on the upper side of the airfoil should not directly influence the solution update at P.



Connectivity points using edge based data structure for an unstructured grid

Figure 2.2: Connectivity definition for nodes from unstructured grid

Hence this region should be excluded or blocked out from the connectivity set for the point P. This region we refer it to as eclipsed region. The pre-processor while generating the connectivty data should take care to eliminate such eclipsed regions. Initially the pre-processor will group all the points contained in the rectangle to form the connectivity data for point P. Then a set of rays are defined, connecting point P to all the points in the connectivity. All the points contained in those rays which intersect the body define the eclipsed region as shown in the figure. The final connectivity is then obtained by deleting the points contained in the eclipsed region. The final connectivity set is also shown in the figure, represented by the rays joining the point P to all the valid nodes in the connectivity.

Let us also now look into some more aspects of the connectivity. Recall that the formulae for F_x and F_y of any function F in 2-dimensions was obtained by minimising the error E as defined below with respect to F_x and F_x respectively.

$$E \equiv \sum_{i=1}^{n} \left(\Delta F_i - F_{x_o} \Delta x_i - F_{y_o} \Delta y_i \right)^2, \tag{2.1}$$

where $\Delta F_i = F_i - F_o$. The least squares formulae for the gradients are

$$\begin{aligned} F_{x_o} &= \frac{\|\Delta y\|^2 (\Delta x, \Delta F) - (\Delta x, \Delta y)(\Delta y, \Delta F)}{\|\Delta x\|^2 \, \|\Delta y\|^2 - (\Delta x, \Delta y)^2} \\ F_{y_o} &= \frac{\|\Delta x\|^2 (\Delta y, \Delta F) - (\Delta x, \Delta y)(\Delta x, \Delta F)}{\|\Delta x\|^2 \, \|\Delta y\|^2 - (\Delta x, \Delta y)^2}, \end{aligned} \tag{2.2}$$

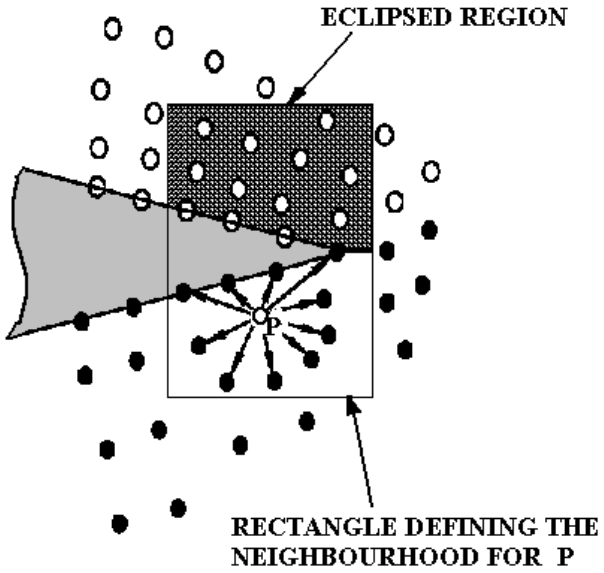


Figure 2.3: Definition of Eclipsed region

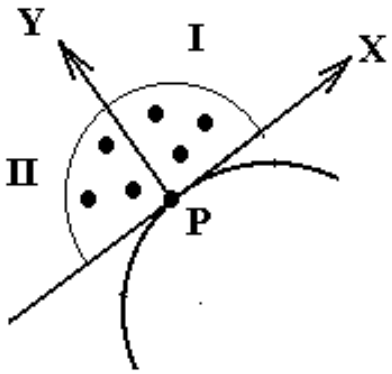


Figure 2.4: Stencil splitting for boundary points

where,

$$\begin{aligned} \|\Delta x\|^2 &= \sum_{i=1}^{n} \Delta x_i^2, \qquad \|\Delta y\|^2 = \sum_{i=1}^{n} \Delta y_i^2 \\ (\Delta x, \Delta F) &= \sum_{i=1}^{n} \Delta x_i \Delta F_i \\ (\Delta y, \Delta F) &= \sum_{i=1}^{n} \Delta y_i \Delta F_i. \end{aligned}$$

here index i represents any point in the connectivity data and n is the total number of nodes in the connectivity data excluding P_o.

In the above approach we can easily see that n should be at least greater than 2 so that we have an overdetermined system for which the least squares minimization procedure given by equation (2.1) becomes meaningful. Also we should notice that the denominator in the formulae for the gradients should not become zero. The denominator can become zero only when all the points in the connectivity lie on a straight line that passes through the node 'o'. Even when points may not lie exactly on a straight line but if they lie approximatly on a line, then also the value of the denominator can sometimes become extremely low leading to solution instabilities. This tendency of the points to lie along a straight line we refer to as degeneracy. Thus while the connectivity data is generated one should ensure that degeneracy is avoided.

In order to evaluate the various split flux derivatives, upwind principle needs that these derivatives be evaluated using only points in the appropriate halves of the plane as shown in fig. 1.2. Thus we can see that the connectivity data should be such that, each half of the plane as shown in fig. 1.2 should contain atleast 3 points (and all the three should not be on a straight line passing through the point $'o'$). Further in case of boundary points, as can be seen from fig. 2.4 only two quadrants are available. In order to evaluate split flux derivatives for the boundary point update, we will have to use points available in the two quadrants. Also in order to satisfy the upwind property we are constrained to use points from a suitable quadrant only. For example consider the updating of any point P lying on a solid boundary as shown in figure2.4. For this case we should ensure that minimum number of points exist in the connectivity for each quadrant and also these points do not form a degenerate case. In general, while generating connectivity for all points including the boundary nodes, we ensure that the connectivity data satisfies the criteria of minimum number of points, no degenracy quadrant wise and avoid points from eclipsed region. The criteria of minimum number of points in the connectivity is satisfied with respect to each quadrant. This way we can ensure that the boundary point update is possible. Also it should be noticed, that by ensuring this minimum number of points quadrant wise, each half of the plane will automatically have double the minimum required ensuring that the interior point update is also problem free.

2.3 Quadtree data structure

As remarked many times before, generation of the proper connectivity data is at the heart of the success of the method. One of the simplest approaches would be to search through all nodes for the nearest number of nodes in each quadrant. However when the number of nodes becomes large, a lot of effort is unneccessarily wasted in the search operations. In order to do these search operations in an optimum way, it would be advisable to use a suitable search algorithm. For the geometric proximity problems like finding a nearest vertex or finding all vertices in a certain range, quadtree data structure is very widely used[76, 26, 84, 85]. The present method infact needs the data of nearest nodes in a small region around each point and hence quadtree data structures are very useful for the connectivity generation. Even though quadtree data structure has been used in many applications, implementation of it differs from problem to problem. We describe here this data structure in view of its application to the Least squares approach. The program using this data structure to generate the connectivity has been developed in C language. Also intelligence has been built in the pre-processor so that the connectivity generated satisfies the desired properties previously mentioned.

Let us first take a quick look into the basic concepts of the quadtree. At the top of the tree, we have a 'root' quadrant(fig. 2.5). This is parent to four children, which in turn may parent each four children. The information of the vertices is found in the quadrants at the lowest level that do not parent children. These terminal quadrants are called 'leaves' or leaf quads.

To start with we have an input file containing the data of all the nodes with flags attached to each node to identify the type of node. For nodes lying on boundaries additional information of the local co-ordinate system is also provided in this input file. The input data for the pre-processor is as follows,

	(co-ordinates)		(direction cosines)		(boundary flag)
Node no	X	Y	dcx	dcy	iflag
1	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
i	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
npts	.	.	.	.	.

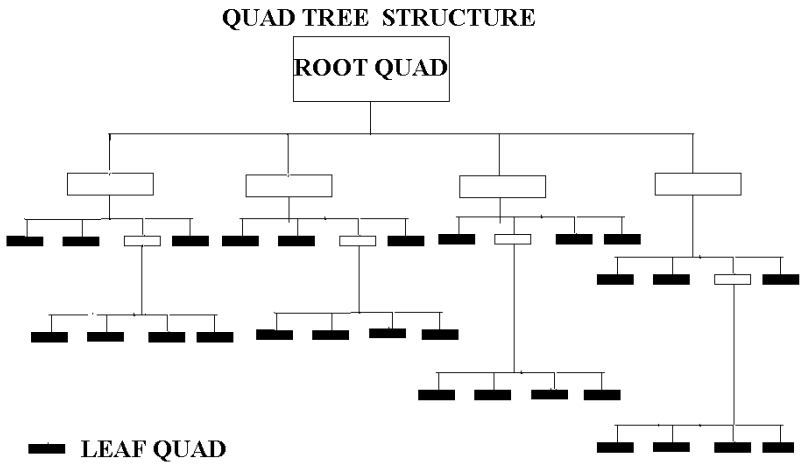


Figure 2.5: quad-tree structure

```
iflag = 1   : point on the solid boundary.
iflag = 2   : point adjacent to the boundaries.
              (solid or outer boundary, in our case we identify a
               point as adjacent to boundary when the connectivity
               contains a node from the boundary)
iflag = 3   : point in the interior.
iflag = 4   : point on the outer boundary.
```

The quad tree structure built around the input data is then stored using the following structures defined as below.

```
struct qtree{
 int isdiv,level,pt[4];
 long int qnum;
 double xq[2],yq[2];
 struct qtree *parent,*child[4];}


struct link{
  long int point_no;
  struct link *next;}
```

The input data is arranged in the quad tree format using above structures. These are made use of in finding the connectivity. The qtree structure stores the following details of the quad tree structure.

```
integer variables
isdiv : = -1 ; if the quad has children
        =  n ; for a leaf quad, n being the number of
               nodes in the quad. n can have a maximum
               value of 4 and the minimum can be zero,
level : = 0 ;  for the root quad and the value reduces every time
               by 1 for every further division of the quad. The exact
               value being dependent on the what level the particular
               quad is.
pt[4] :        This integer array contains the addresses of the nodes
               contained in the leaf quad. If the isdiv value is -1 then
               they are initialsed to null values as this quad would then
               not directly contain any nodes, but is only further
               subdivided into four children
qnum  :        This variable stores the quad number. The qnum for root
               quad is 1. Every time a quad is further subdivided new
               numbers are assigned to these quads.

real variables

xq[2],yq[2] :  They store the values of the coordinates of the quad
               as shown in figure(2.6)

qtree type pointer variables

parent      :  This contains the pointer to the parent quad. For the
               root quad it will point to NULL.
child[4]    :  This pointer array contains the address of the four children
               quad. In case of a leaf quad all the array elements are made
               to point to NULL.
```

The link structure is a linked list of vertices defined for all points. This list contains the connectivity data. Following are the details stored in this structure.

```
Integer variable
point_no  : Address of the node
link type pointer variable
next      : Pointer to the next node in the linked list.
```

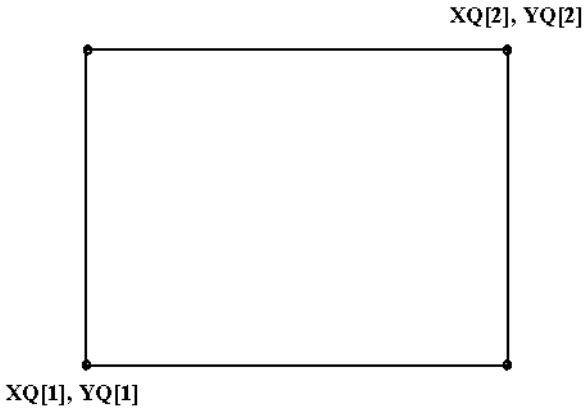


Figure 2.6: Storage of quad coordinates

The address of the first node in the link structure is the point itself for which the connectivity set is defined. The "next" pointer for the last node in the linked list will point to NULL. A pictorial representation of this is shown in figure2.7.

Connectivity generation is done using Traverse algorithm and Range search algorithm. Given any node, traverse algorithm is used in locating the leaf quad in which the node is located. This basically consists of moving down the quad tree structure from the root quad until one reaches the leaf quad in which it lies. Based on the size of the quad in which the node is located, a Range box is defined around the leaf quad. Figure 2.8 shows a typical quad structure with the range box defined around a leaf quad. The size of the Range box is usually a multiple of the quad size in which the node of interest is located. Using Range search algorithm, typically involving searching for all the quads intersecting with the range box, all the nodes lying in the quads which intersect the range box are arranged in a link list. This forms the basic or the raw connectivity for the node. The link list of vertices is then passed through several checks, so that the connectivity generated satisfies the various constraints described before. The constraints are, minimum number of points in each quadrant, non-occurence of degeneracy and effective blocking of eclipsed regions. If the constraints are not satisfied then, the range box is expanded and the procedure of creating the link list of nodes is repeated until the connectivity generated satisfies the constraints imposed.

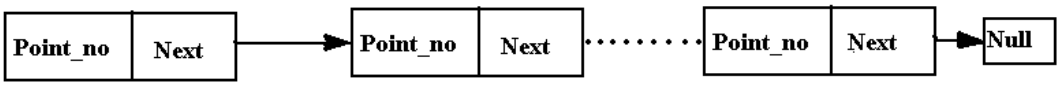


Figure 2.7: Link list structure for connectivity data

2.4 Application of quadtree pre-processor to LSKUM

In this section we give details about the application of the quadtree pre-processor to connectivity generation on a variety of 2-D distributution of points. In our present work we have considered point distributions obtained from a variety of methods. These distributions are mainly derived from the following type of grid generators,

- unstructured grid
- body fitted meshes or structured grids
- Cartesian grids
- multiple overlapping grids(chimera meshes)

In general the procedure followed for the connectivity generations is as follows. First we obtain the basic or the raw point distribution from any one or a combination of the grid generators mentioned above. The node data is then arranged as an input file in the format the pre-processor requires. The preprocessor then operates on these set of nodes and gives the connectivity data as the output. The pre-processor as already mentioned takes care to generate the connectivity satisfying all the constraints imposed. The total number of points in the connectivity(including all the four quadrants for interior points, or the two quadrants for boundary points)is controlled by the user. The pre-processor usually generates the connectivity with a user specified minimum number of points in each quadrant. Based on this minimum number the connectivity data is generated. In exceptional cases, the required minimum number of points in each quadrant may not be available. In such cases a warning message

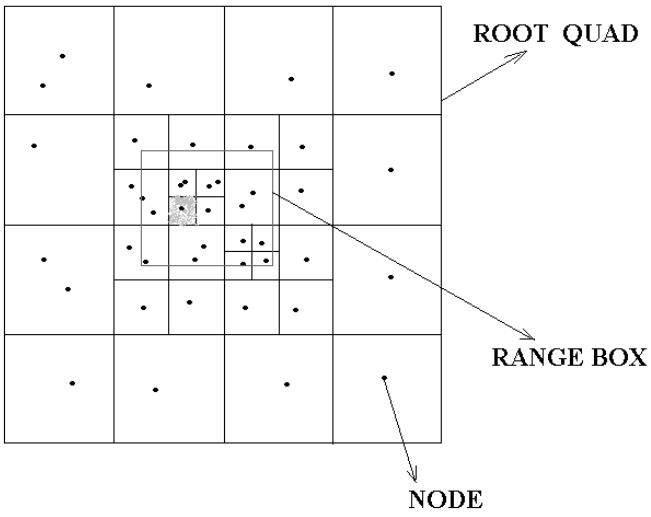


Figure 2.8: Range search

is given and some alternative measures are taken. Such cases usually occur near the boundaries. Consider the point distribution near a solid boundary as shown in fig. 2.9. In this case we can clearly see that quadrant I does not have any points in it. In such cases the pre-processor recognises that for this particular point P, quadrant I is empty. Once it recognises this, the minimum number of points required in each quadrant for the remaining ones are increased. This ensures that at least the required number of nodes are present in each half and this would be sufficient for obtaining least squares approximation of split flux derivatives. A similar case can also occur at the outer boundaries. Consider a point distribution obtained from an unstructured grid as shown in fig. 2.10. In this case we can easily see that the quadrant II has only one point. In such cases the number of points in other quadrants are enhanced so that the required number of nodes in each half is ensured. Detection of the deficient number of nodes and the subsequent remedial measures are all automatically done by the pre-processor. Apart from these cases of deficient number points in certain quadrants near boundaries, there can also be some cases of defective point distribution in the connectivity of an interior node. Consider a point distribution obtained by an overlapping structured and cartesian grid as shown in fig. 2.11. We can observe from the figure that the basic connectivity data does satisfy the criterion of minimum number of three points in each quadrant(excluding points on the axis). However a closer examination of the distribution of points reveals that the points in quadrant I and IV are almost in a vertical line(ie along y-axis). This peculair distribution of points in general poses difficulties in evaluation of x- derivatives of split fluxes. This

difficulty in evaluation of x-derivatives is due to the degeneracy referred to before in section 2.2. The difficulty can be easily explained by noting that the variations in x-coordinate for the points in quadrants I and IV are very minimal and thus would lead to nearly illconditioned linear equations for derivatives. In such cases the connectivity can be easily enhanced to contain more points in the quads I and IV as shown in the figure. The pre-processor automatically notices such defects and makes modifications to the basic connectivity.

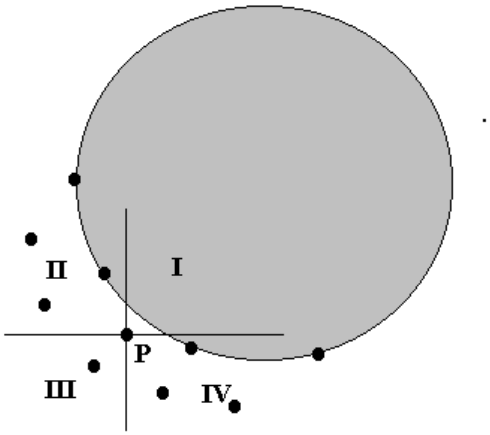


Figure 2.9: Node adjacent to solid boundary with deficient connectivity

Finally we give some examples of connectivity generation using the preprocessor. Figure 2.12 shows a distribution of points around an airfoil obtained using a structured grid. The pre-processor operates on these set of points to generate the connectivity. The preprocessor generates a background quadtree around these points. Using this background quadtree the connectivity of all the nodes is generated. Figure 2.12 also shows the background quadtree with the connectivity(points shown by circles) for a point P on the body. For such points as well as those adjacent to the body, the connectivity should be such that it should not contain nodes from the eclipsed region, as this would be aerodynamically wrong. In our present work, the link list of points generated from the range search is passed through several check routines which deletes all the points in the eclipsed region. Figure 2.13 shows a typical point distribution near the trailing edge of the airfoil. This is a distribution obtained from an unstructured grid. In this figure P is a point on the trailing edge. The connectivity for this point can also be seen in the figure. For this particular point it can be noticed that the connectivity points(indicated by circles) are there on both the upper and lower sides

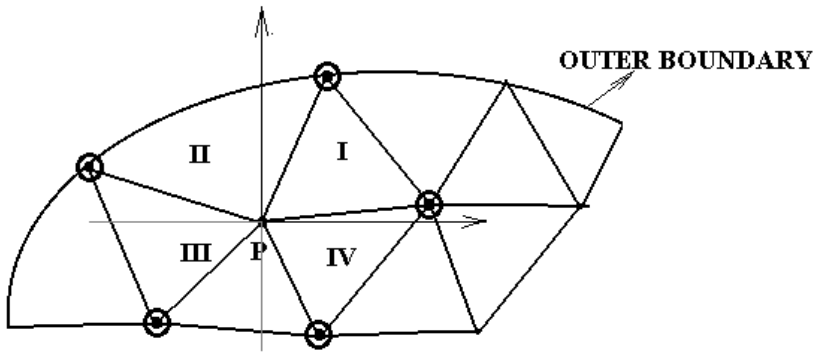


Figure 2.10: Node adjacent to outer boundary with defecient connectivity

of the airfoil. Figure 2.14 shows a point P on the airfoil just ahead of the trailing edge with its connectivty(points indicated by circles). For this case it can be clearly seen there are no points from the undesirable eclipsed regions. As a further illustration, fig. 2.15 shows the connectivity for a point P lying on the outer boundary. The background quads which facilitate the connectivity generation can also be seen in the figure.

The quadtree preprocessor is an essential tool for the connectivity generation for any type of distribution of points. In this chapter we have clearly demonstrated the ability of the preprocessor to generate the connectivity by considering typical examples. In the next chapter we present the results for 2-D LSKUM in combination with the preprocessor for a wide range of problems using a variety of point distributions.

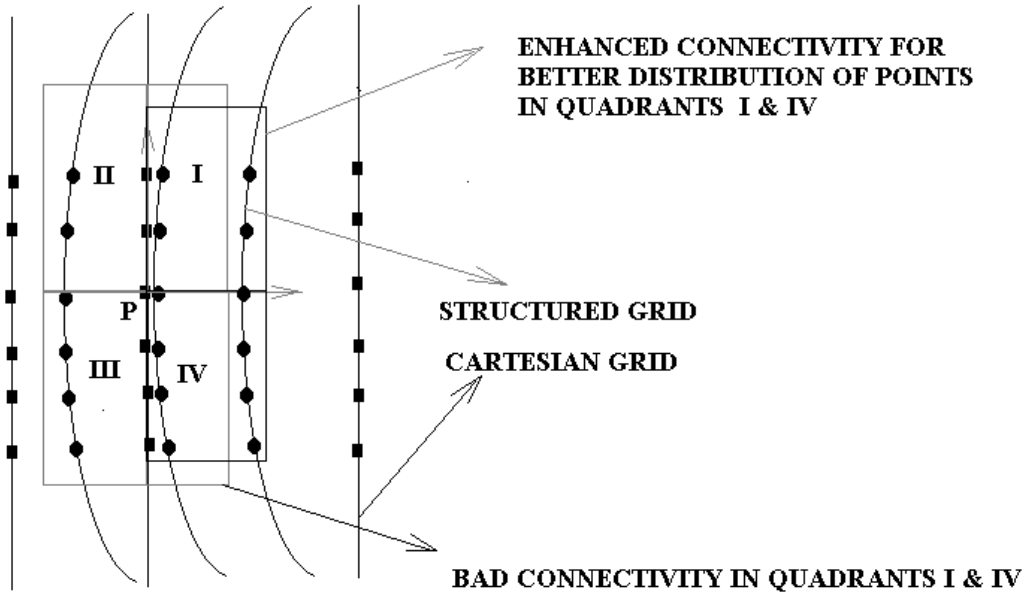


Figure 2.11: Node in an overlapping region with defective connectivity

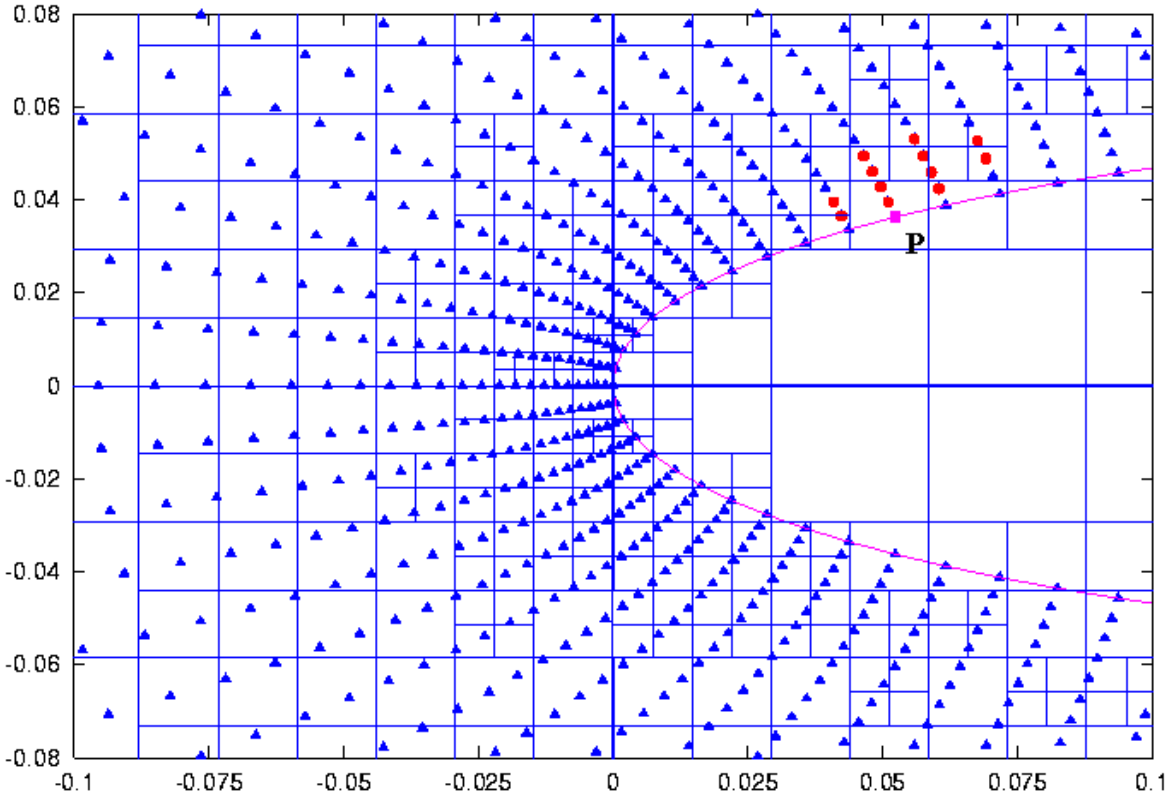


Figure 2.12: Connectivity for a solid boundary node : Points from structured grid

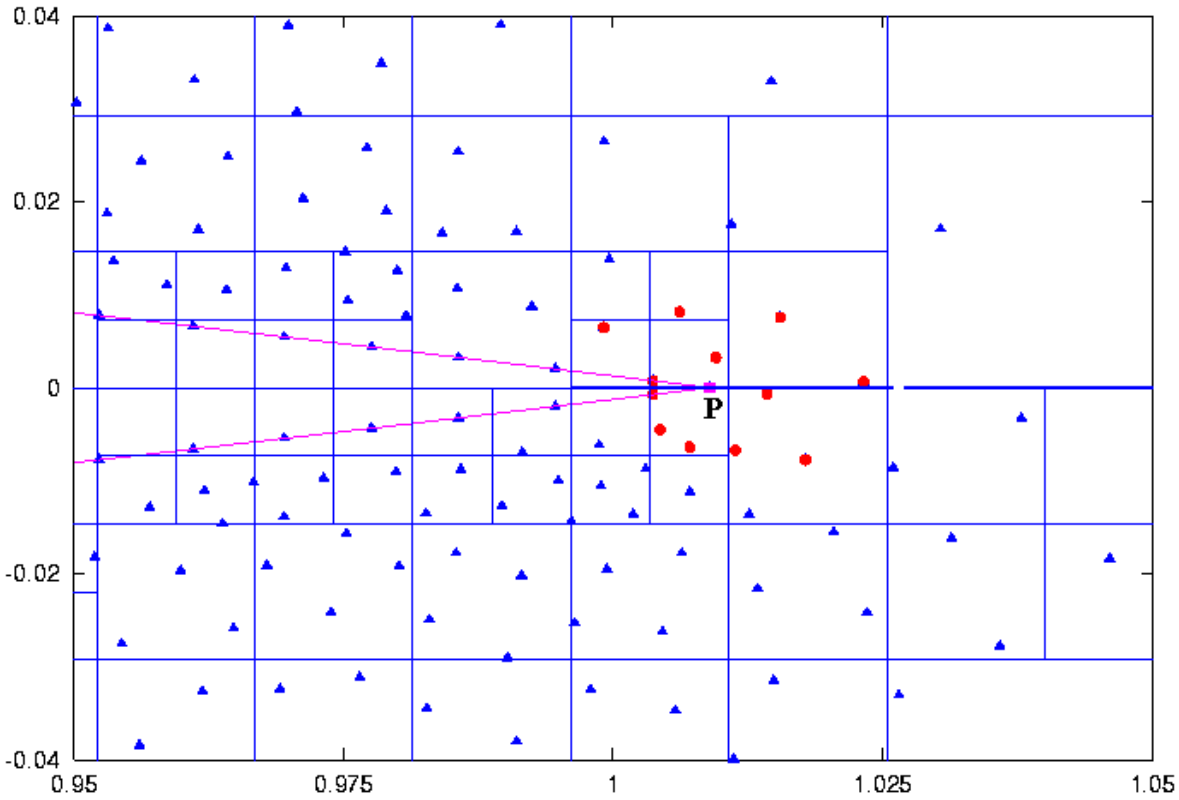


Figure 2.13: Connectivity for trailing edge node : Points from unstructured grid

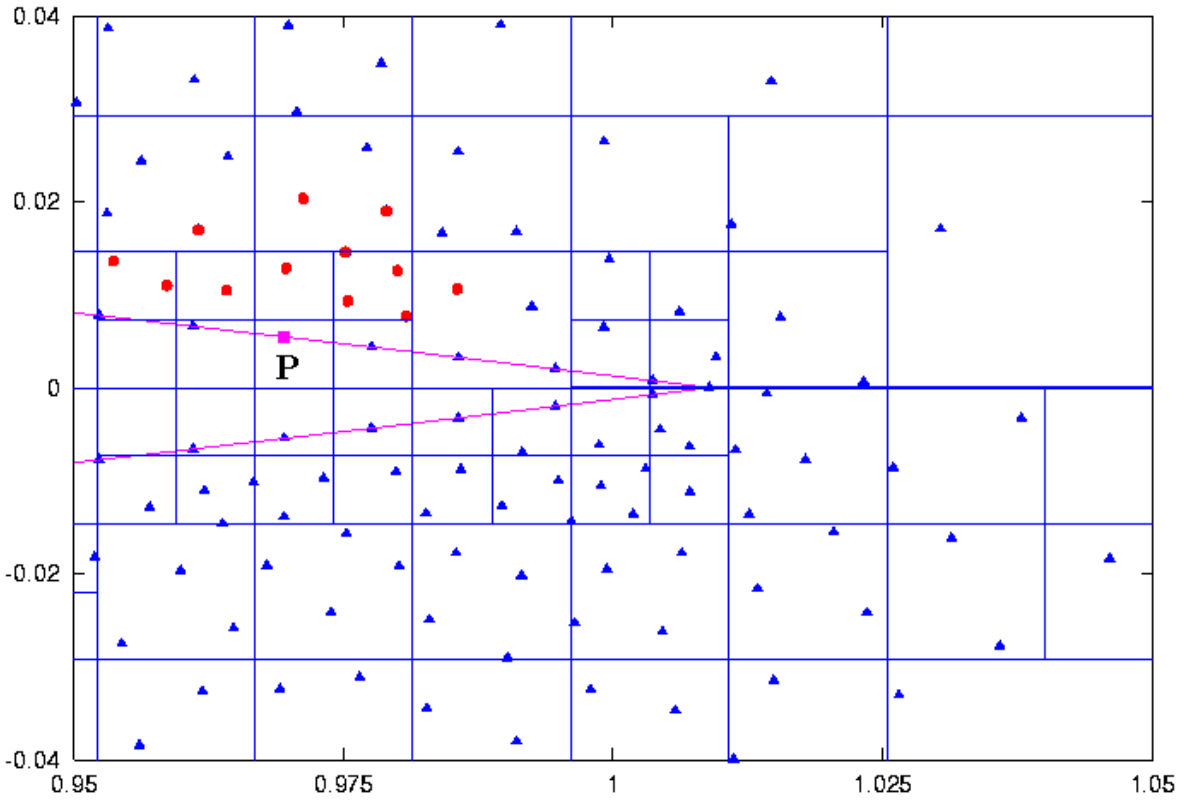


Figure 2.14: Connectivity for a node near TE: Points from unstructured grid

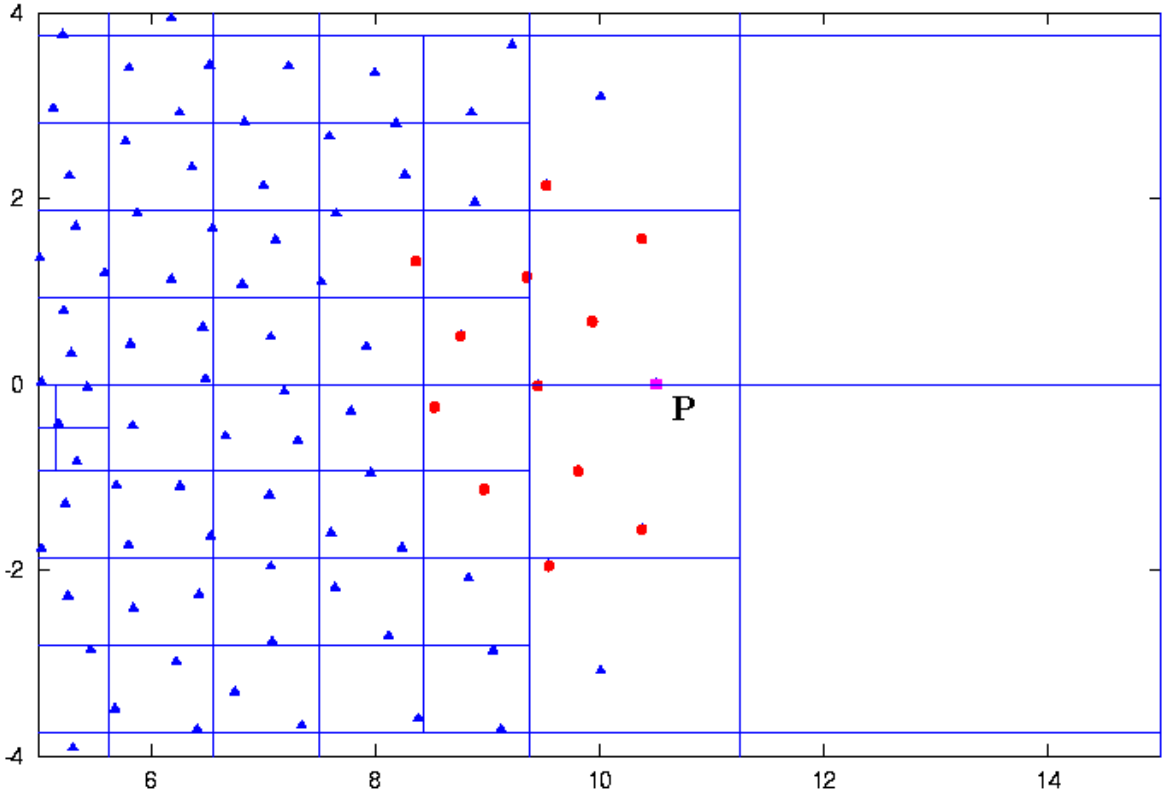


Figure 2.15: Connectivity for a node on the outer boundary: Points from unstructured grid

Chapter 3

Results for 2-D LSKUM

The main purpose of this chapter is to show the capability of LSKUM to work on different types of distribution of points that is, fully exploit its grid free nature. As described in the previous chapters LSKUM solver can operate on any of distribution of points provided it has the proper connectivity data. In the present work we get this crucial connectivity data for any arbitrary distribution of points using the quadtree pre-processor described in the previous chapter. In the following sections we demonstrate the power of LSKUM solver combined with quad tree preprocessor. Towards this purpose we have chosen a variety of 2D flow problems which represent standard GAMM[33] and AGARD[77] test cases. Also we have selected various distributions of points for these test flow problems. The details of these are given in the next section. The important issue we want to highlight here is, even though LSKUM has been shown to work for 2-D cases[29, 30, 31], for the first time we demonstrate the power of LSKUM to work on any distribution of points in conjunction with the quadtree pre-processor. Another aspect of the present work is the use of the Kinetic Outer Boundary Condition(KOBC) for all the computations. This treatment of outer boundary condition here is different from the previous work[30, 31] where Reimann boundary conditions have been used.

3.1 Test Cases

We have chosen the following 2D test cases for computations of flow past NACA0012 airfoil.

- Subsonic flow : $M = 0.63, \alpha = 2.0$
- Transonic flow : $M = 0.85, \alpha = 1.0$
- Supersonic flow : $M = 1.20, \alpha = 0.0$
- Transonic flow past NACA 0012 airfoil Biplane configuration
  $M = 0.85, \alpha = 0.0$

The first three test cases are quite popular in CFD literature. The first one(subsonic case) has $C_d = 0$ and the last one(supersonic case) has $C_l = 0$. Also, the transonic case has a shock on the upper surface and one on the lower surface. Thus both qualitatively and quantitatively they are ideal for assesing the performance of a numerical scheme. The fourth one is also a transonic case for a multielement airfoil.

For the above mentioned flow problems, we obtain the point distributions obtained from various methods. These include structured/unstructured/Cartesian/Overlapping grids. We have also point distributions obtained from overlapping grids. The details of these are as metioned below.

3.1.1 Point distribution from structured O-grid

This distribution consists of 9600 nodes with 160 nodes on the body and and 60 nodes in the normal direction. This is obtained using an O-type grid. The outer boundary is located approximatley at 10 chord distance. For such a distribution of points, as previously discussed one can use a simple pre-processor to generate the connectivity. However this pre-processor would then be very specific. In the present work we use the general purpose quadtree preprocessor described earlier to generate the connectivity. Figure 3.1 shows a view of the cloud of points around the airfoil.

3.1.2 Point distribution from unstructured grid

In this case we have a set of 4074 nodes with 160 points on the body and 40 points on the outer boundary. The outer boundary in this case is also appoximately located at 10 chord distance. The nodes for this case are obtained from an unstructured grid generator using Delaunay triangulation. The pre-processor then operates on this set of points to generate the connectivity data. Again in this case it is possible to have a pre-processor specific to these type of distibution of points, using the edge based data structure to generate the connectivity. In the present work as mentioned before we have used the general purpose quadtree pre-processor. Figure 3.2 shows a view of the distribution of points around the airfoil for this case.

3.1.3 Point distribution from embedded Cartesian grids

In this case we obtain the point distribution from various levels of embedded cartesian meshes. For this application we define the airfoil by a set of points distributed independently on the body, rather than defining the body by the points formed by the intersection of the Cartesian gridlines with the airfoil. For this case we have two slight variants as follows.

1. Points from embedded Cartesian grids only

2. Points from embedded Cartesian grids together with an overlapping tiny polar mesh at the trailing edge(in some ways similar to the FAME idea[1]).

The total number of nodes in the first case without the polar mesh at the trailing edge is 19112. The second grid with polar mesh at trailing edge has 19321 points. Also the airfoil is independently defined by a set of 160 nodes without considering the intersection of the cartesian lines with the body. However all the nodes which lie inside the body are blanked. The outer boundary is located at 8 chord distance from the body. Figure 3.3 shows a typical view of the embedded cartesian grid. One can observe the different grid point spacing for each of the embedded grid. Figure 3.4 shows the connectivity for a point in the transition region. Also the background quads which facilitates the connectivity generation can also be seen in the figure. Even though the point distribution pattern changes at the transition from finer to coarser grid the solver does not face any problem. Figure 3.5 further shows a closer view of the grid with the overlapping polar mesh at the trailing edge.

3.1.4 Point distribution from structured O-grid overlapping with Cartesian grid

This set of points is obtained from a body fitted O-mesh overlapping with Cartesian grid. The body fitted mesh around the airfoil extends approximately upto 5 chord distance. Over this grid we have an overlapping cartesian grid which extends upto 10 chord distance towards the outer boundary. The cartesian mesh does not extend upto to the body in the interior. It stops at about 1 chord distance from the body. The total number of points is 12884. There are 160 points on the body obtained from the body fitted mesh. On the outer boundary there are 440 points obtained from the Cartesian grid. Figure 3.6 shows the plot for this case. One can observe that the the two grids have an overlapping region which starts at about 1 chord distance away from the airfoil and extends upto about 5 chord distance from the airfoil.

3.1.5 Point distribution from chimera mesh containing two structured O-Grids with background Cartesian mesh

In this section we consider a biplane configuration consisting of two NACA0012 airfoils. The two airfoils are seperated by half chord distance vertically as well as horizantally as shown in figure 3.7. The set of points for this case is obtained from grids consisting of two overlapping body fitted O-grids around each airfoil approximately upto 1.5 chord distance and from a background Cartesian grid. On both the airfoils there are 160 nodes obtained from the body fitted grids. The outer boundary is located approximatly at 7 chord distance. The total number of grid points is 19144. Figure 3.7 shows a typical view of the point distribution for this case. This grid is an example of what we call multiple overlapping grids. The cloud of points obtained is due to these mutually overlapping grids.

In the present study we have been able to operate the solver on all these types of distributions without having to make any changes or adjustments in the flowsolver.

This has been possible because the solver has the capability to operate on any arbitrary distribution of points. The main input the solver needs is the connectivity data. In our present work full power of the LSKUM is realised along with the quadtree preprocessor which generates the connectivity data for any arbitrary distribution of points. At this juncture we also want to emphasize that, our aim here is to demonstrate the power of the solver to operate on arbitrary distribution of points rather than concentrating on getting very accurate results(these can always be obtained by mesh refinement).

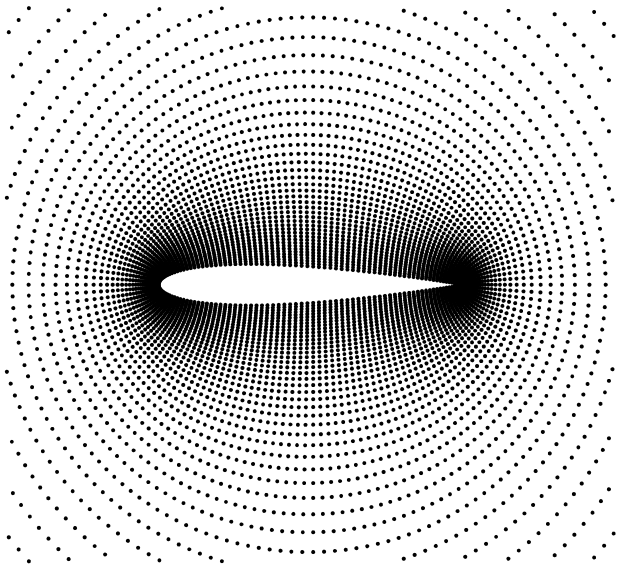


Figure 3.1: A view of the point distribution from structured Grid

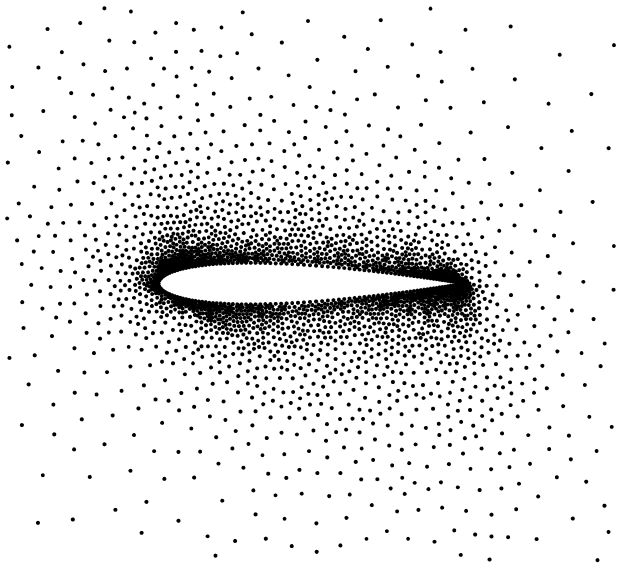


Figure 3.2: A view of the point distribution from unstructured Grid

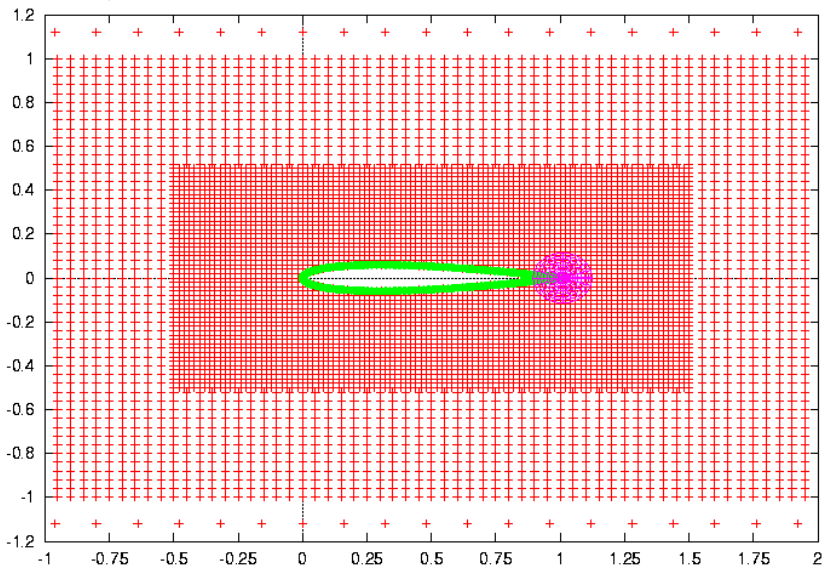


Figure 3.3: A view of the point distibution from embedded Cartesian Grids

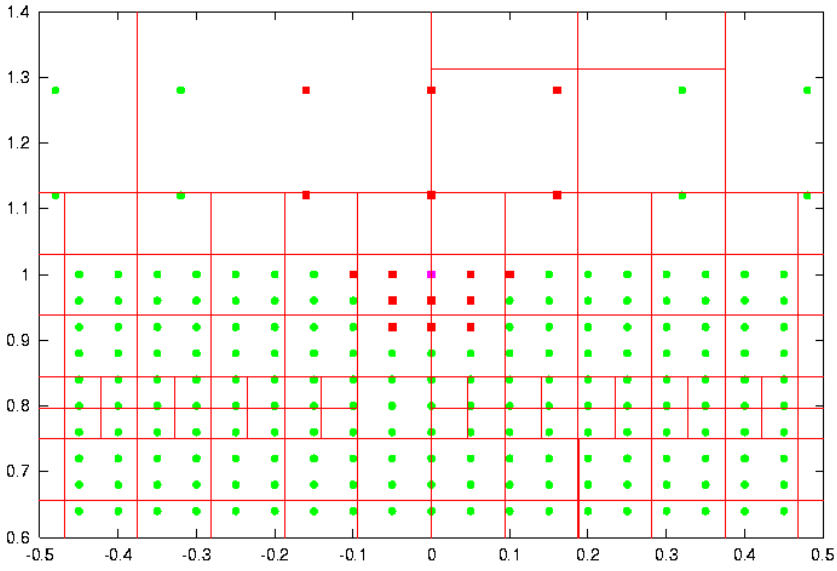


Figure 3.4: A view of the connectivity in transition region

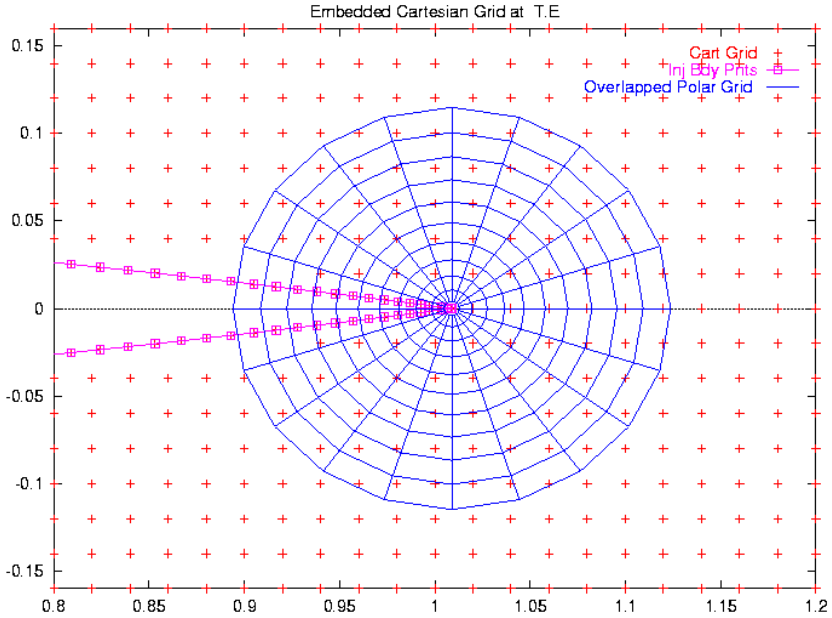


Figure 3.5: A view of the point distribution for overlapped polar grid at T.E

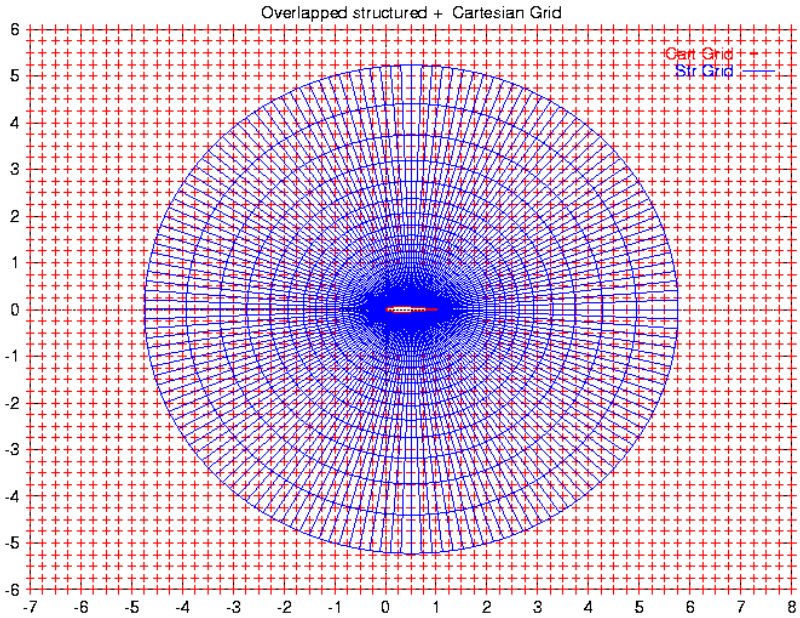


Figure 3.6: A view of the point distribution from overlapping structured and Cartesian grid

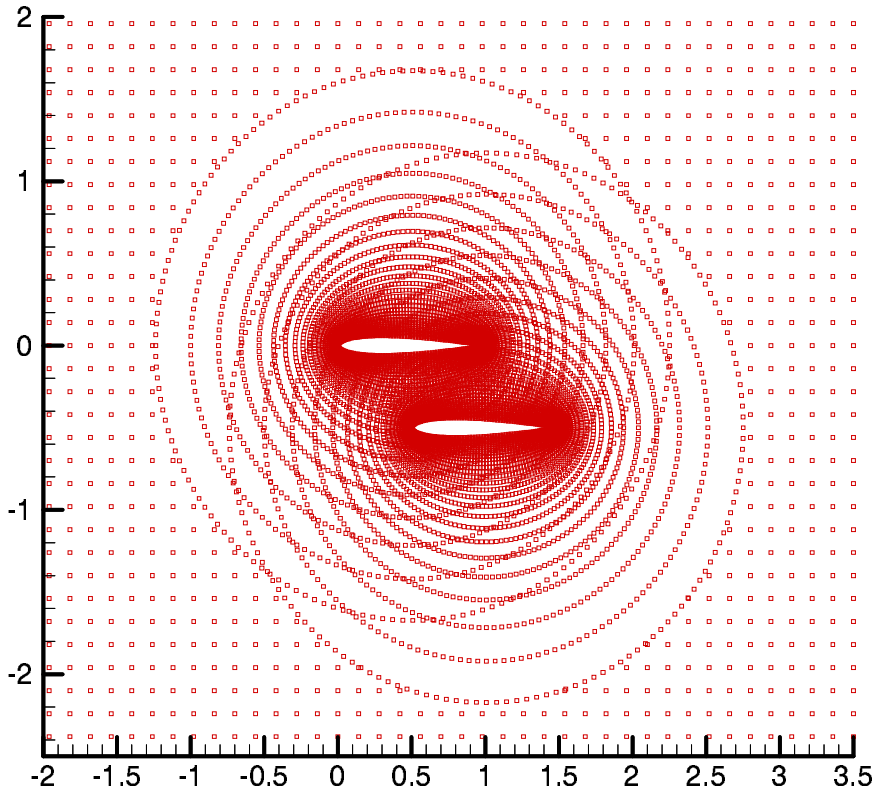


Figure 3.7: A view of the point distribution from overlappped grids for NACA0012 Biplane

3.2 Results on points distribution from structured O-Grid

Figures 3.8 and 3.9 show the residue plots and convergence history of C_l and C_d for the subsonic cases. Figure 3.10 shows the pressure coefficient plot and the pressure contours are shown in fig. 3.11. All the results indicate the flow features such as leading edge suction, pressure on the upper and lower surface and the smoothness of pressure contours have been captured. Table 3.1 shows a comparison of the computed results with the GAMM values. A good comparison has been obtained. Similar plots for the transonic case are shown in figures 3.12 to 3.15 and table 3.2 shows the comparison of computed results with the AGARD results. All the important features such as shock location on upper and lower surface are captured well. Further C_l and C_d are satisfactorily predicted. Similar comparisons have been made for the supersonic case also. Various plots for this case can be seen in figures 3.16 to 3.19. Finally the comparisons of predicted C_l and C_d are shown in table 3.3. The two values of the residual rmax and rss are defined as

$$\text{rmax} = \max(\delta_n), \quad \delta_n = \frac{\rho^{n+1} - \rho^n}{\rho^n} \tag{3.1}$$

$$\text{rss} = \sqrt{\frac{1}{np} \sum_{i=1}^{n} \delta_n^2} \tag{3.2}$$

where np represents the total number of nodes. The values plotted in the convergence history are the logarithm of rmax and rss.

Grid Type	C_l	C_d
Structured Grid	0.316	0.004
GAMM	0.329-0.336	0.3E-04-0.7E-3

Table 3.1: C_l C_d : Structured O-Grid $M = 0.63$ $\alpha = 2.0^o$

Grid Type	C_l	C_d
Structured Grid	0.3305	0.058
AGARD	0.330-0.389	0.0464-0.0590

Table 3.2: C_l C_d:Structured O-Grid:$M = 0.85$ $\alpha = 1.0^o$

Grid Type	C_l	C_d
Structured Grid	0.00016	.097
AGARD	should be 0	0.0946-0.096

Table 3.3: C_l C_d :Structured O-Grid:$M = 1.20$ $\alpha = 0.0^o$

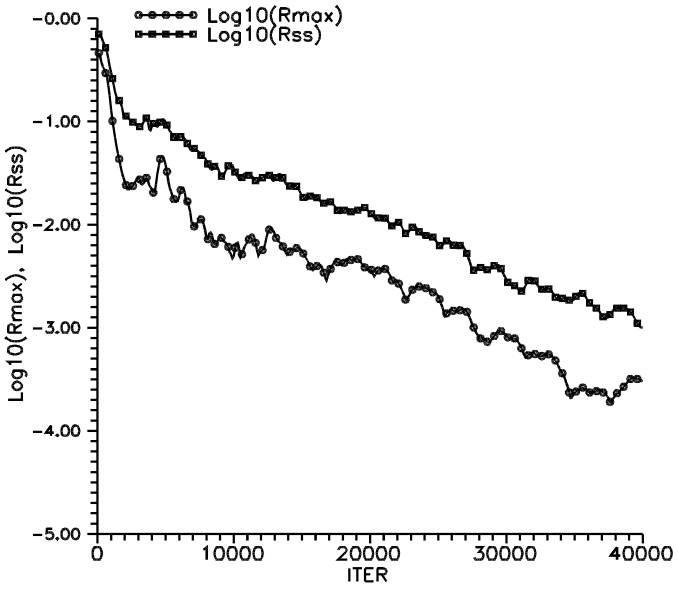


Figure 3.8: Residue Plots : Structured O-grid :$M = 0.63 \;\; \alpha = 2^o$

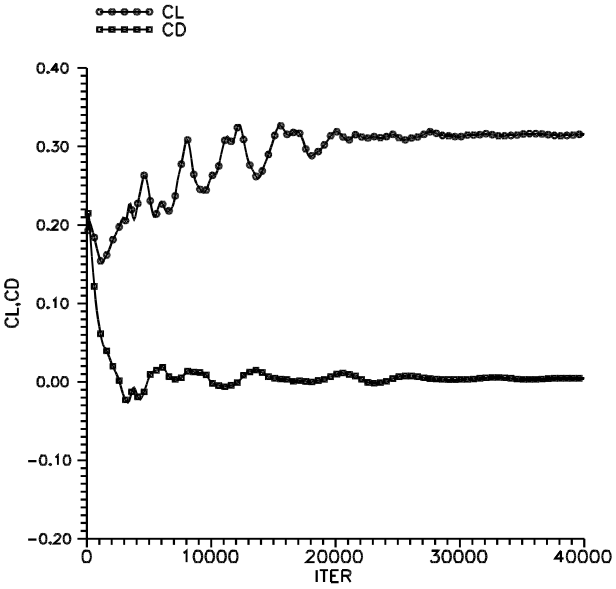


Figure 3.9: C_l C_d convergence: Structured O-grid :$M = 0.63 \;\; \alpha = 2^o$

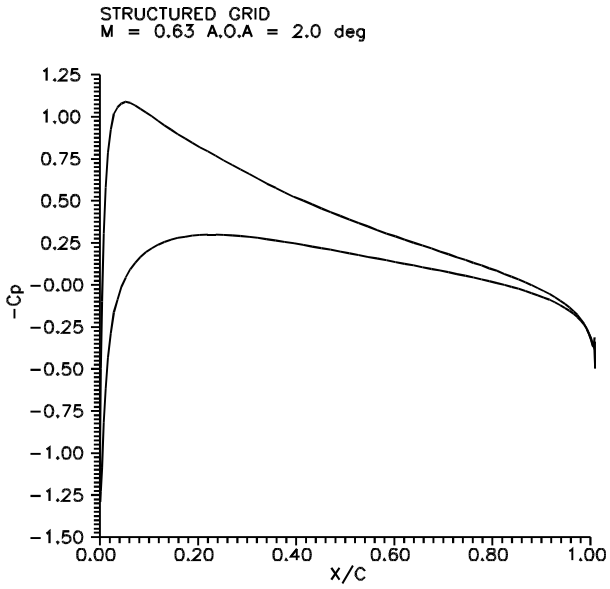


Figure 3.10: Cp Plots : $M = 0.63 \;\; \alpha = 2^o$

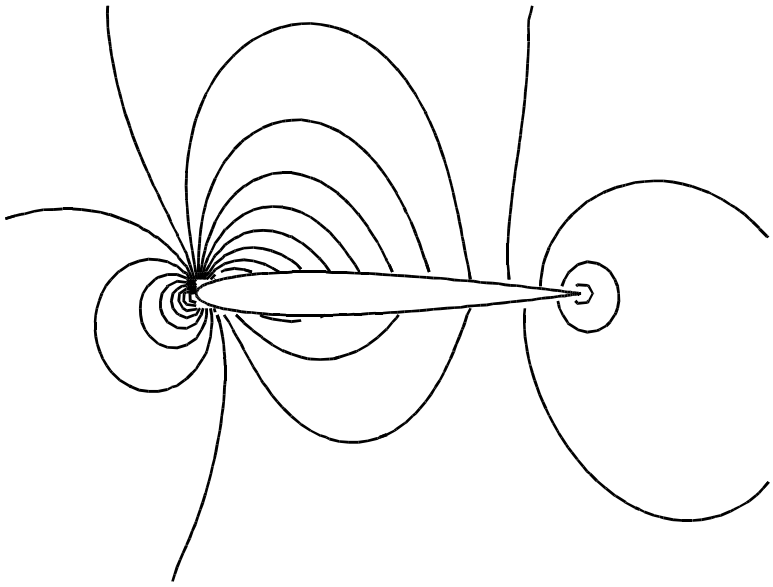


Figure 3.11: Pressure contours: Structured O-grid : $M = 0.63 \;\; \alpha = 2^o$

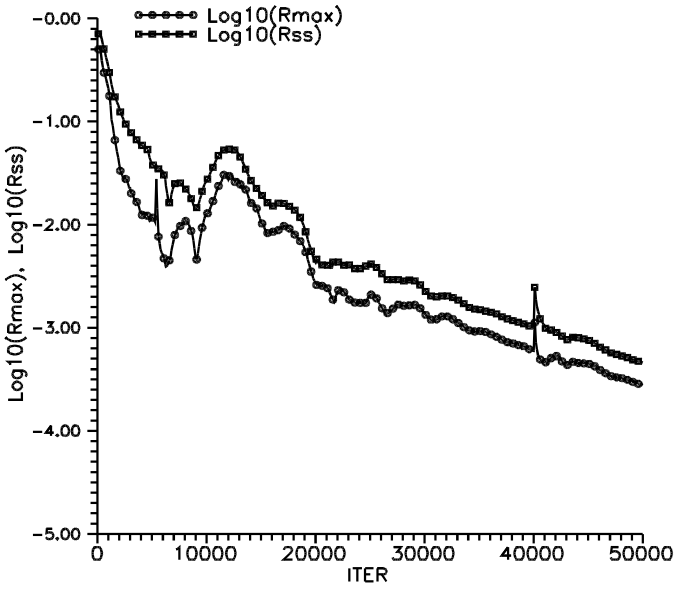


Figure 3.12: Residue Plots: Structured O-grid :$M = 0.85 \;\; \alpha = 1^o$

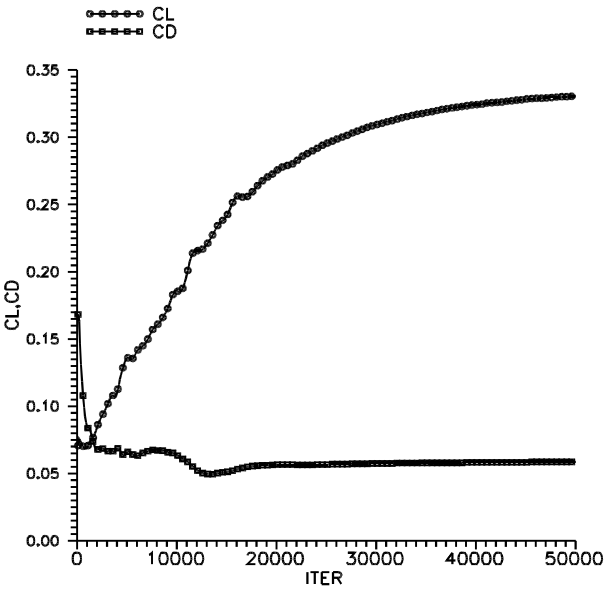


Figure 3.13: C_l C_d convergence: Structured O-grid :$M = 0.85 \;\; \alpha = 1^o$

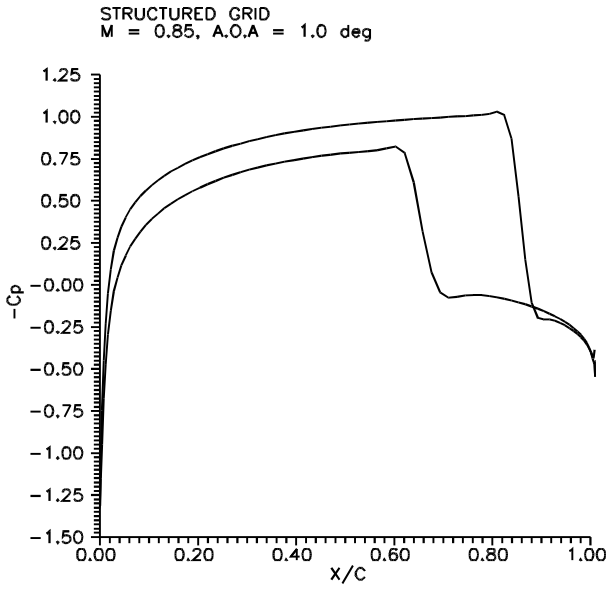


Figure 3.14: Cp Plots :$M = 0.85 \;\; \alpha = 1^o$

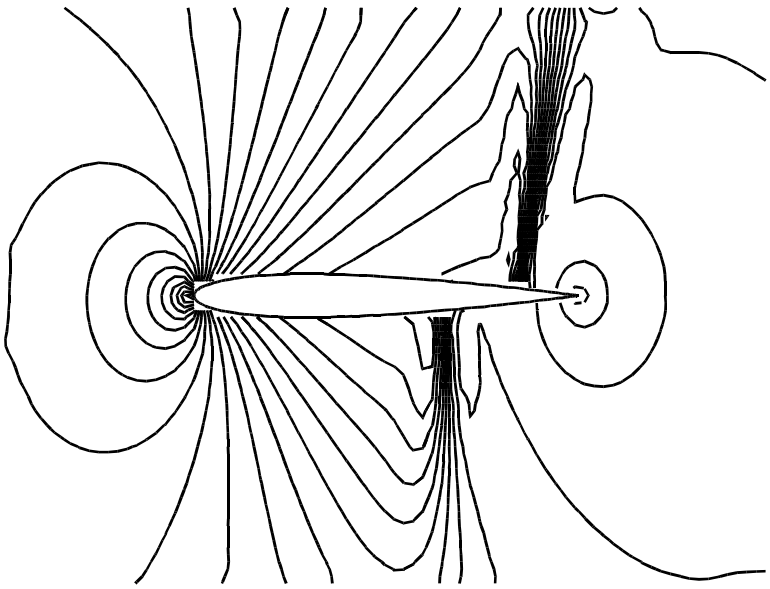


Figure 3.15: Pressure contours : Structured O-grid : $M = 0.85 \;\; \alpha = 1^o$

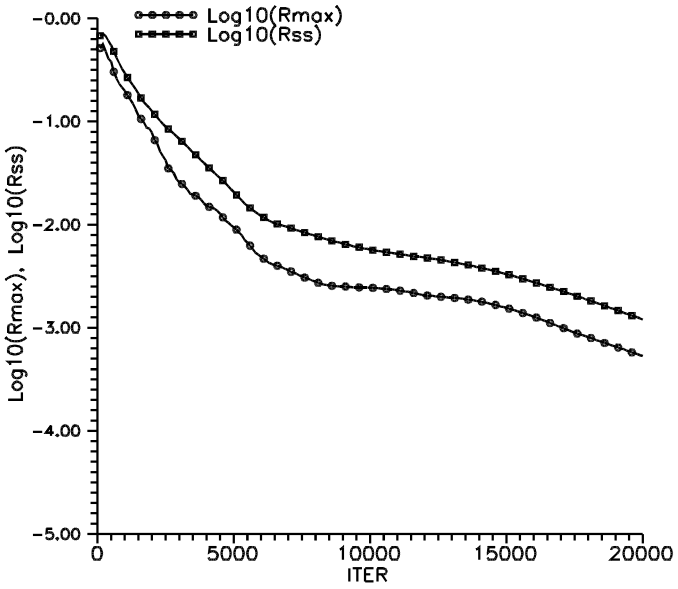


Figure 3.16: Residue Plots: Structured O-grid : $M = 1.20 \;\; \alpha = 0^o$

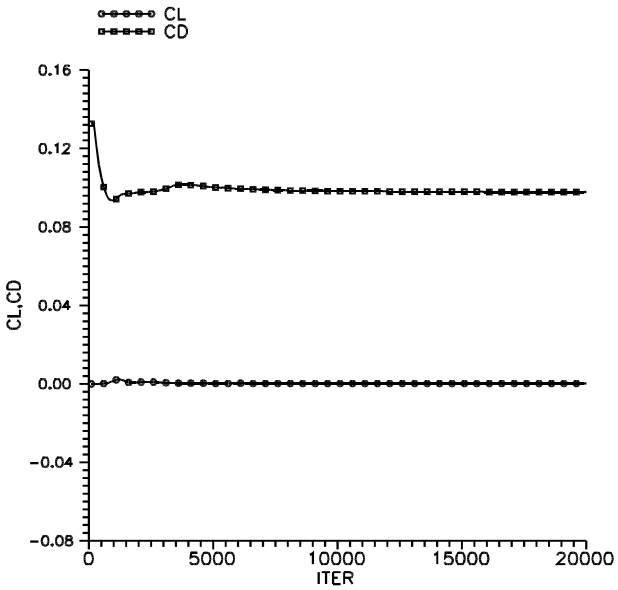


Figure 3.17: $C_l \; C_d$ convergence: Structured O-grid : $M = 1.20 \;\; \alpha = 0^o$

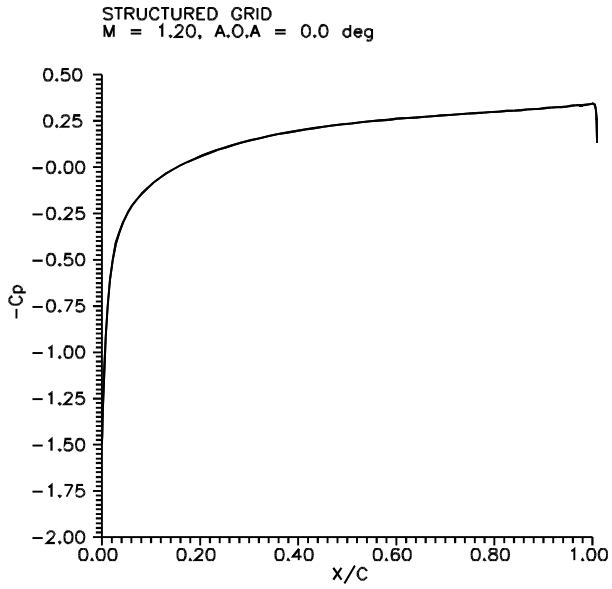


Figure 3.18: Cp Plots : $M = 1.20$ $\alpha = 0^o$

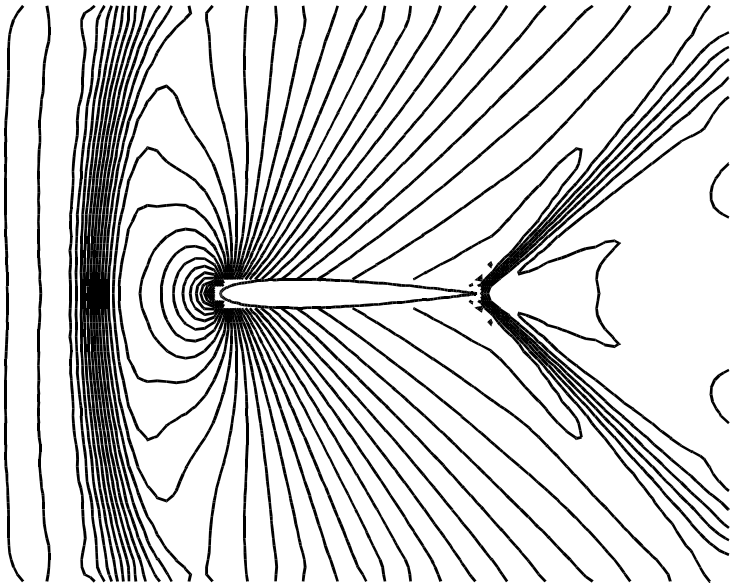


Figure 3.19: Pressure contours: Structured O-grid : $M = 1.20$ $\alpha = 0^o$

3.3 Results on points distribution from unstructured Grid

Figures 3.20 and 3.21 show the residue plots and convergence histories of C_l and C_d for the subsonic cases. Figure 3.22 shows the pressure coefficient plot and the pressure contours are shown in fig. 3.23. All the results indicate the flow features have been satisfactorily captured. Table 3.4 shows a comparison of the computed results with the GAMM values. A good comparison has been obtained. Similar plots for the transonic case can be seen in figures 3.24 to 3.27 and table 3.5 shows the comparison of computed results with the AGARD results. All the important features such as shock location, C_l and C_d are satisfactorily predicted. Similar comparisons have been obtained for the supersonic case also. Various plots for this case can be seen in figures 3.28 to 3.31. Finally the comparison of predicted C_l and C_d are shown in table 3.6. Overall results show that the LSKUM preprocessor combination performs on unstructured grid as well as for the structured grid case. As mentioned before, the primary purpose of this study is to show the power of LSKUM and preprocessor combination. It is always possible to obtain more accurate results by mesh refinement. For example, the C_l for the supersonic case can be made zero more accurately(more number of zeros before the first significant digit) by increasing the number of nodes.

Grid Type	C_l	C_d
Untructured Grid	0.320	0.007
GAMM	0.329-0.336	0.3E-04-0.7E-3

Table 3.4: C_l C_d : Unstructured Grid $M = 0.63$ $\alpha = 2.0^o$

Grid Type	C_l	C_d
Unstructured Grid	0.378	0.062
AGARD	0.330-0.389	0.0464-0.0590

Table 3.5: C_l C_d : Unstructured Grid:$M = 0.85$ $\alpha = 1.0^o$

Grid Type	C_l	C_d
Unstructured Grid	-0.007	.099
AGARD	should be 0	0.0946-0.096

Table 3.6: C_l C_d: Unstructured Grid:$M = 1.20$ $\alpha = 0.0^o$

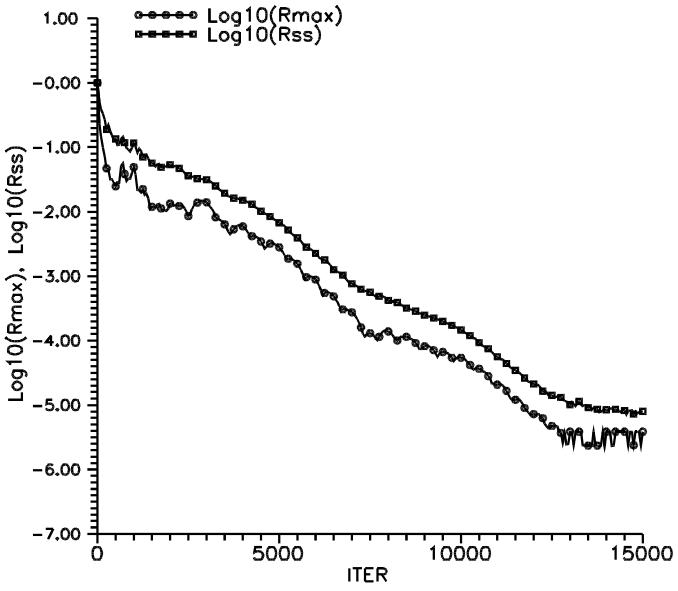


Figure 3.20: Residue Plots : Unstructured grid :$M = 0.63 \;\; \alpha = 2^o$

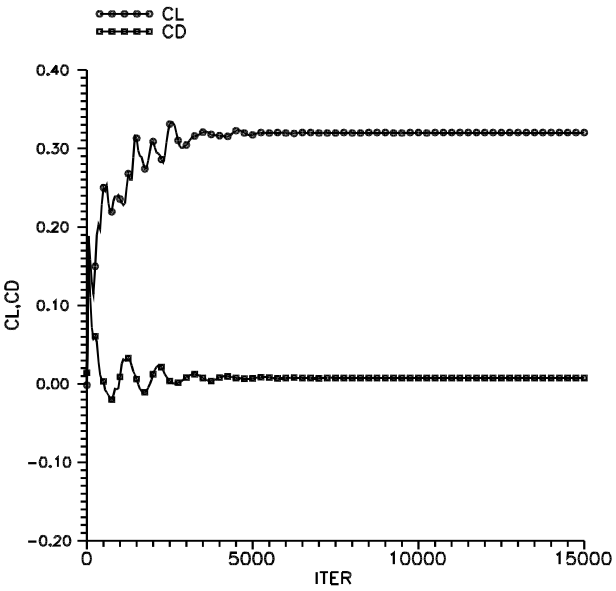


Figure 3.21: C_l C_d convergence: Unstructured grid :$M = 0.63 \;\; \alpha = 2^o$

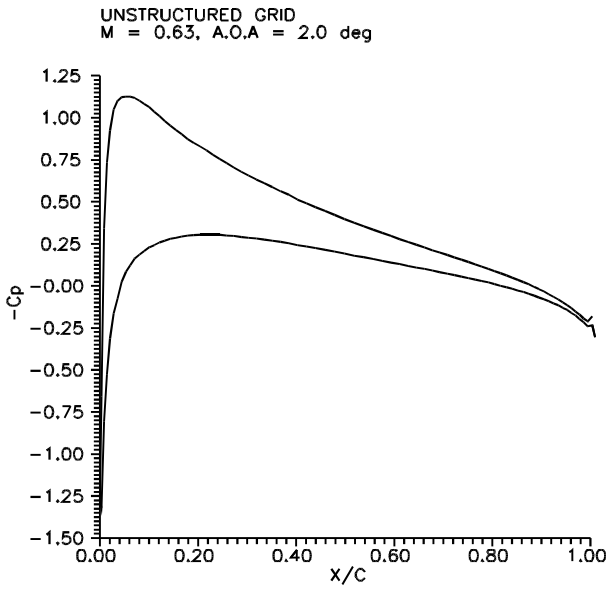


Figure 3.22: Cp Plots : $M = 0.63 \;\; \alpha = 2^o$

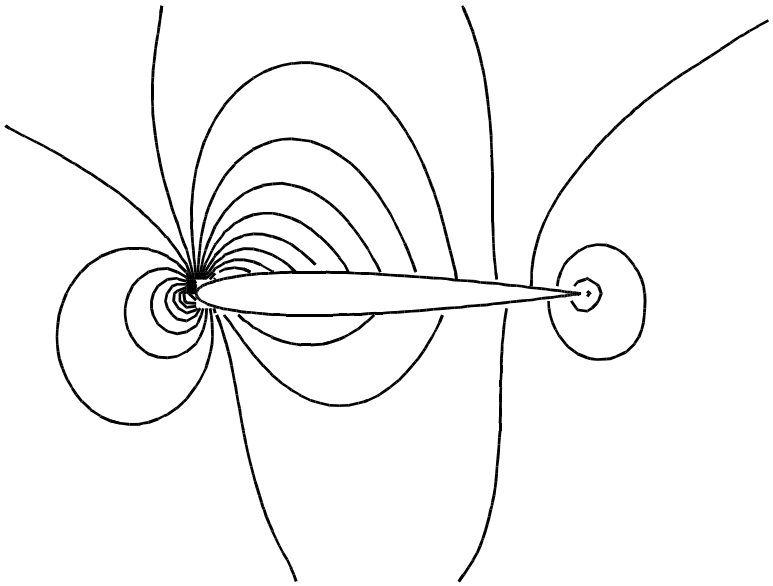


Figure 3.23: Pressure contours: Unstructured grid : $M = 0.63 \;\; \alpha = 2^o$

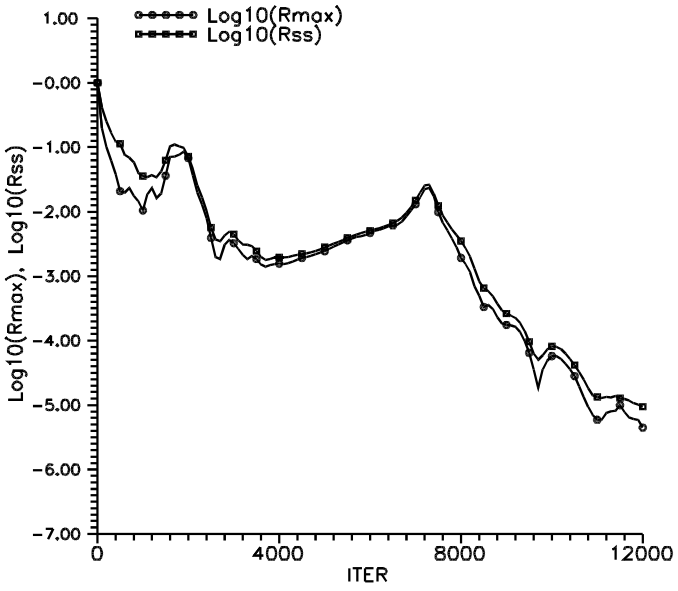


Figure 3.24: Residue Plots: Unstructured grid :$M = 0.85$ $\alpha = 1^o$

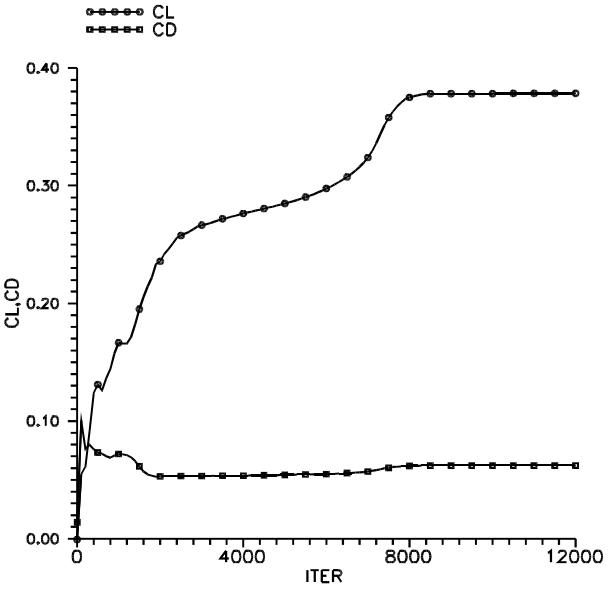


Figure 3.25: C_l C_d convergence: Unstructured grid :$M = 0.85$ $\alpha = 1^o$

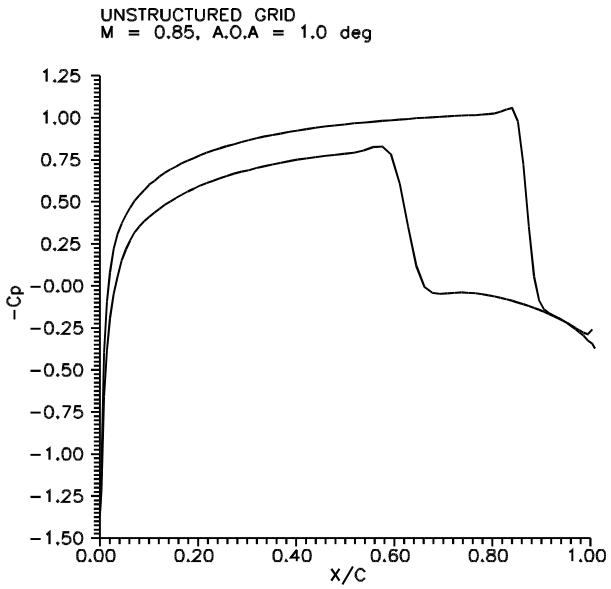


Figure 3.26: Cp Plots :$M = 0.85 \;\; \alpha = 1^o$

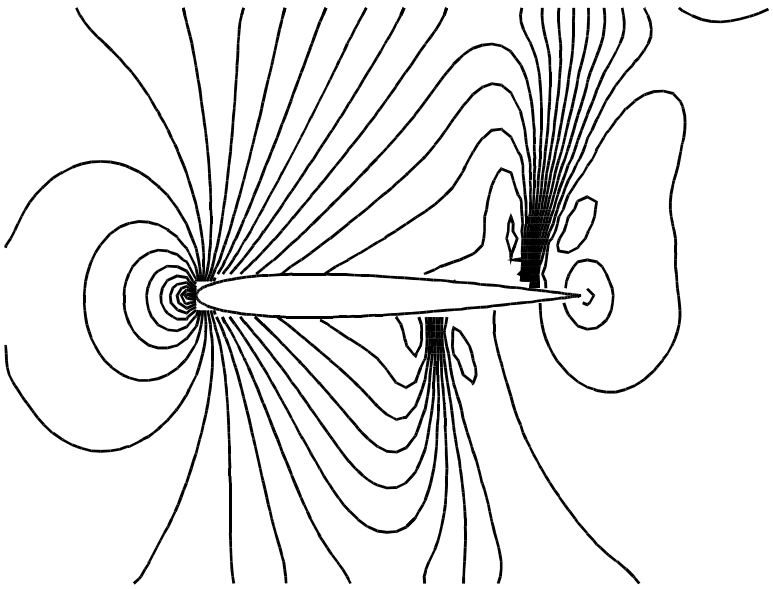


Figure 3.27: Pressure contours: Unstructured grid : $M = 0.85 \;\; \alpha = 1^o$

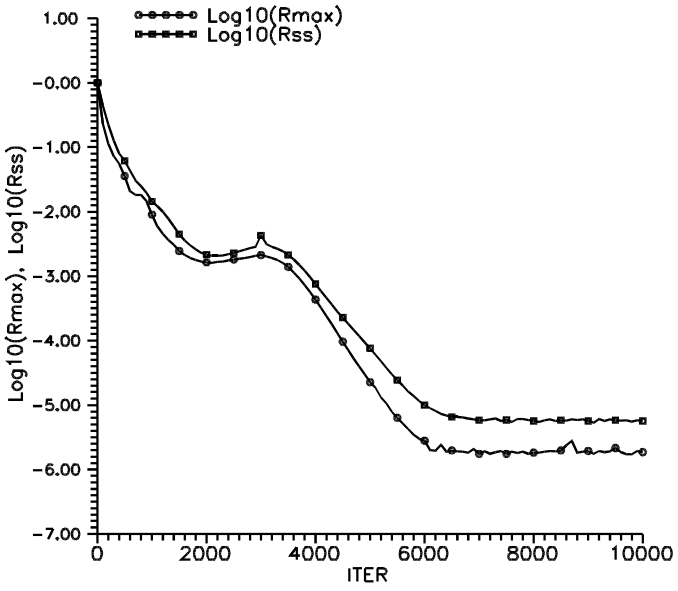


Figure 3.28: Residue Plots: Unstructured grid : $M = 1.20 \;\; \alpha = 0^o$

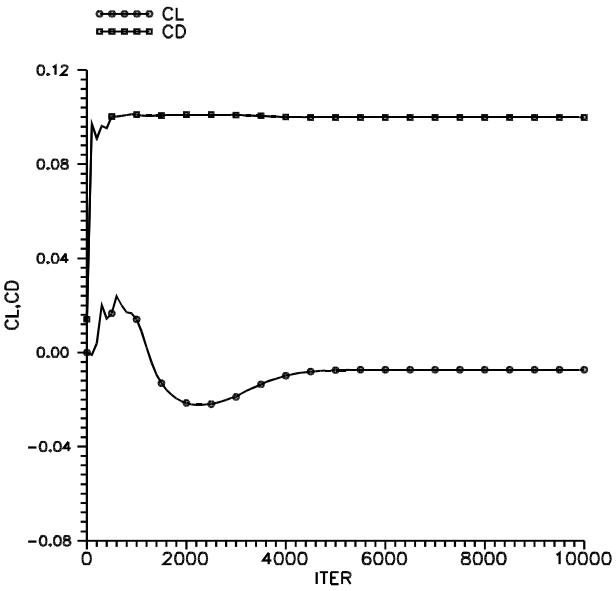


Figure 3.29: C_l C_d convergence: Unstructured grid : $M = 1.20 \;\; \alpha = 0^o$

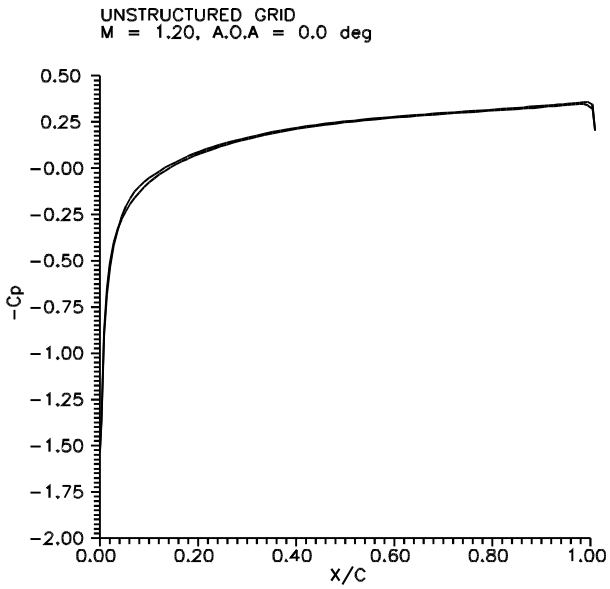


Figure 3.30: Cp Plots : $M = 1.20 \;\; \alpha = 0^o$

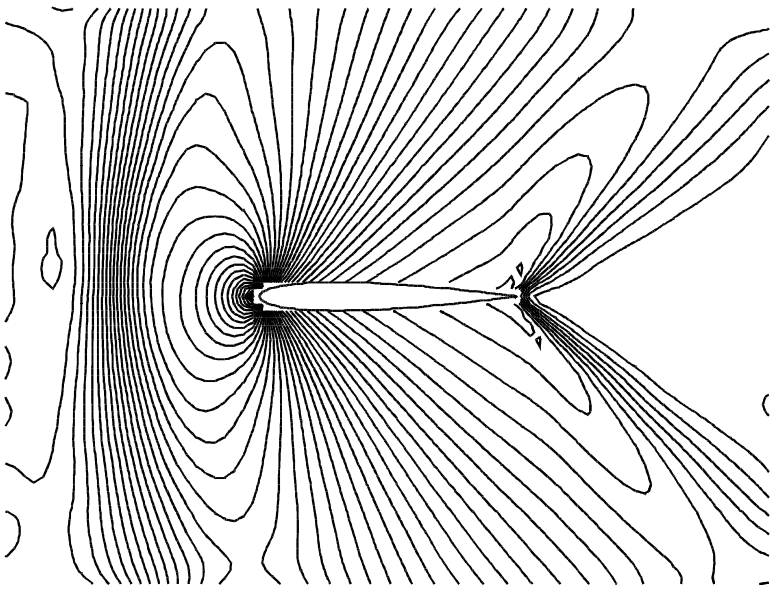


Figure 3.31: Pressure contours:Unstructured grid : $M = 1.20 \;\; \alpha = 0^o$

3.4 Results on points distribution from embedded cartesian Grid

For this point distribution we have results for two cases, with and without the small overlapping polar grid at the trailing edge. Figures 3.32 and 3.33 show the residue plots and the C_l and C_d history for the embedded cartesian grids with and without the polar grid at the trailing edge. The polar grid overlaps with the cartesian grid over the airfoil upto about 10% of the chord length. These plots are for the subsonic case $M = 0.63, \alpha = 2.0^o$. The maximum residue(rmax) as well the root mean square value of the residue(rss) have been shown in the residue plots for all the cases. It can be seen that the solution converges faster in the case where the small polar grid is present. The reason for this is that, in the case of the polar grid at the trailing edge we are able to run the solver at higher CFL number. For the case of embedded grid alone, higher CFL number poses instability problem at the trailing edge. Figure 3.34 and figure 3.35 show the density contours at the trailing edge for this grid with and without the polar grid at the trailing edge. We can clearly see that the overlapping polar grid resolves the flow more smoothly near the trailing edge which is clearly brought out by the smoother density contours for the latter case. Figure 3.36 and figure 3.37 show the pressure contours over the airfoil and the C_p distribution over the airfoil respectively for the embedded cartesian grid. Figure 3.38 and figure 3.39 show similar plots for the Embedded grid with the polar grid at trailing edge. For both these cases we can observe that the overall prediction is quite similar. The C_p distribution for the latter case is smoother in the trailing edge region. Table 3.7 shows a comparison of the predicted C_l and C_d values with the GAMM values. All the predicted values are in the vicinity of the GAMM values. However with the polar grid the C_l is slightly reduced. In general using Cartesian grids alone it is not possible to get very accurate results as the geometry will not be adeqautely resolved . However in the present study we have addressed this issue to some extent by using different levels of embedded cartesian grids. Also the airfoil is defined by points distributed independently on the body.

Grid Type	C_l	C_d
Embedded cartesian Grid	0.312	0.026
Embedded cart. grid + Polar grid	0.280	0.021
GAMM	0.329-0.336	0.3E-04-0.7E-3

Table 3.7: C_l C_d :embedded cartesian grid $M = 0.63$ $\alpha = 2.0^o$

For the transonic case, fig. 3.40 and fig. 3.41 show the residue plots and the C_l, C_d history for the embedded cartesian grids with and without the polar grid at the trailing edge. The convergence history is more or less the same for both the cases. Figure 3.42 and figure 3.43 show the pressure contours over the airfoil and the pressure

coefficient distribution over the airfoil respectively for the embedded cartesian grid alone. Figure 3.44 and fig. 3.45 show similar plots for the Embedded grid with the overlapping polar grid at trailing edge. For both these cases we can observe that the overall prediction is quite similar. The Cp distribution near the traling edge region for the latter case is smoother in the trailing edge region. Table 3.8 shows a comparison of the predicted C_l and C_d values with the AGARD values.

Grid Type	C_l	C_d
Embedded cartesian Grid	0.302	0.077
Embedded cart. grid + Polar grid	0.262	0.071
AGARD	0.330-0.389	0.0464-0.0590

Table 3.8: C_l C_d: embedded cartesian grid : $M = 0.85$ $\alpha = 1.0^o$

For the supersonic case, fig. 3.46 and fig. 3.47 show the residue plots and the C_l, C_d history for the embedded cartesian grids with and without the polar grid at the trailing edge. The residue fall for the polar grid case is slightly more compared to the cases of embedded grid cartesian grid without polar grid at trailing edge. However both the cases show a similar trend for the C_l and C_d convergence. Figure 3.48 and fig. 3.49 shows the pressure contours over the airfoil and the pressure coefficient distribution over the airfoil respectively for the embedded cartesian grid. Figure 3.50 and fig. 3.51 shows similar plots for the Embedded grid with the polar grid at trailing edge . For both these cases we can observe that the overall prediction is quite similar. The pressure coefficient distribution for both the cases are almost similar. The lift and drag co-efficients are almost of the same value. Table 3.9 shows a comparison of the predicted C_l and C_d values with the AGARD values. It can be observed from the table that the computed values are in reasonable agreement with the AGARD range.

Grid Type	C_l	C_d
Embedded cartesian Grid	0.001	0.11
Embedded cart. grid + Polar grid	0.001	0.108
AGARD	should be 0	0.0946-0.096

Table 3.9: C_l C_d: embedded cartesian grid : $M = 1.20$ $\alpha = 0.0^o$

Referring to tables 3.7, 3.8 and 3.9 we can observe that for all the cases we obtain reasonable results that can be expected out of cartesian grids. In all the cases the solutions obtained with the polar grid shows a smoother pressure distribution near the trailing edge. We can also observe that the computed C_l are on lower side for this case. Perhaps this is due to the limitations of pure cartesian grids to resolve the flow accurately near the body. Therefore it would be desirable that if we could have a

point distribution from a body fitted mesh near the body and then use a background cartesian mesh away from the body. In the next section we discuss the results for computations on such a type of point distribution.

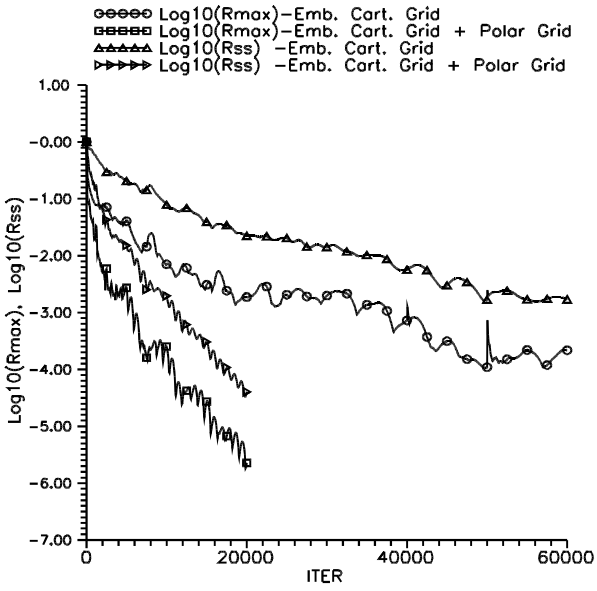


Figure 3.32: Residue Plots - embedded cartesian grid $M = 0.63 \;\; \alpha = 2^o$

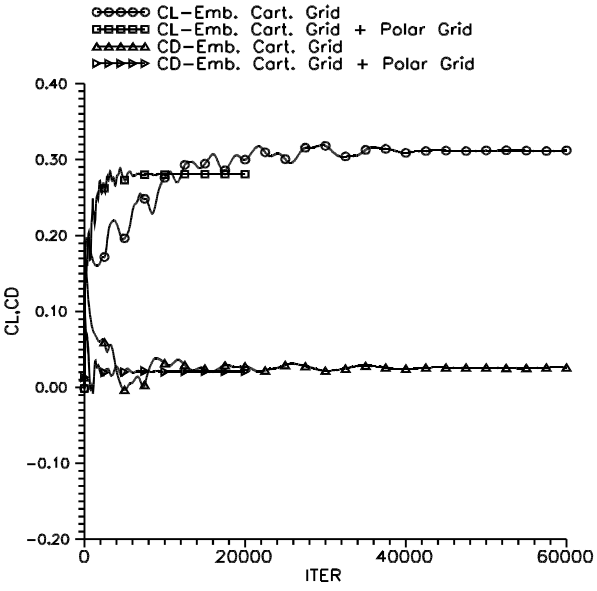


Figure 3.33: C_l C_d convergence - embedded cartesian grid$M = 0.63 \;\; \alpha = 2^o$

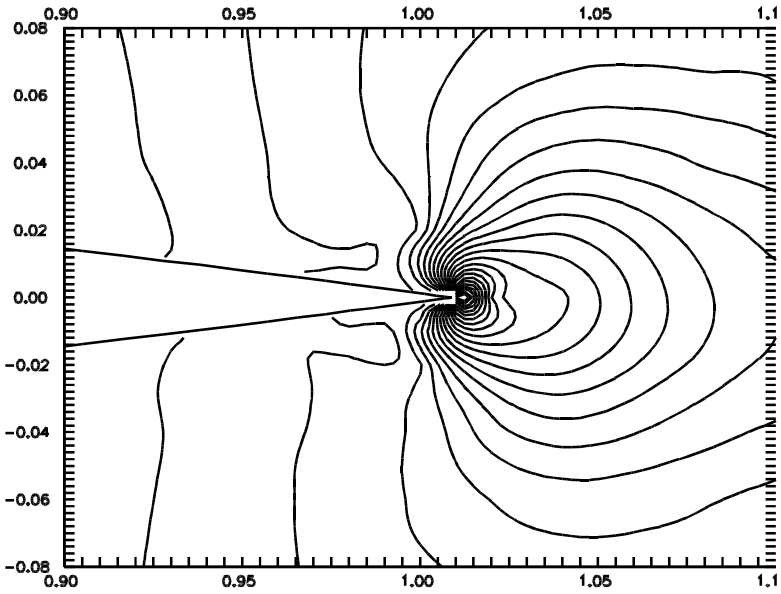


Figure 3.34: Density contours for embedded cartesian grid

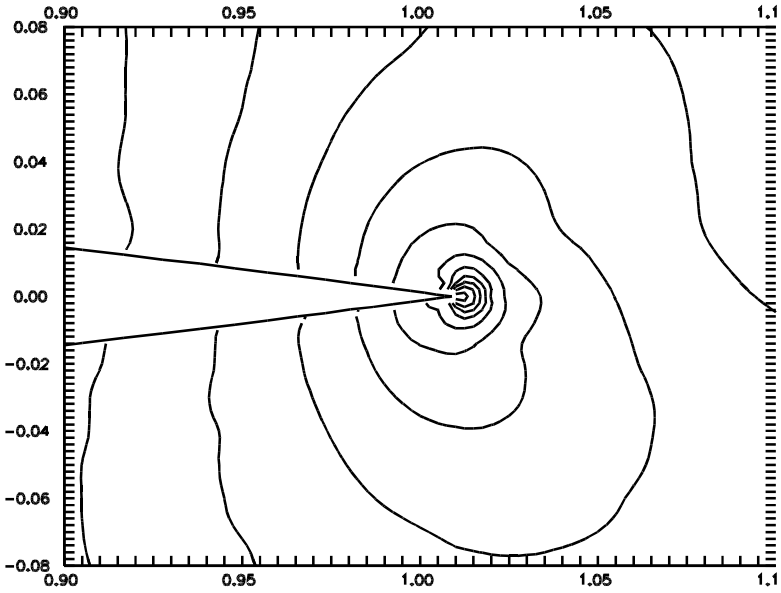


Figure 3.35: Density contours for embedded cartesian grid + Polar grid at T.E

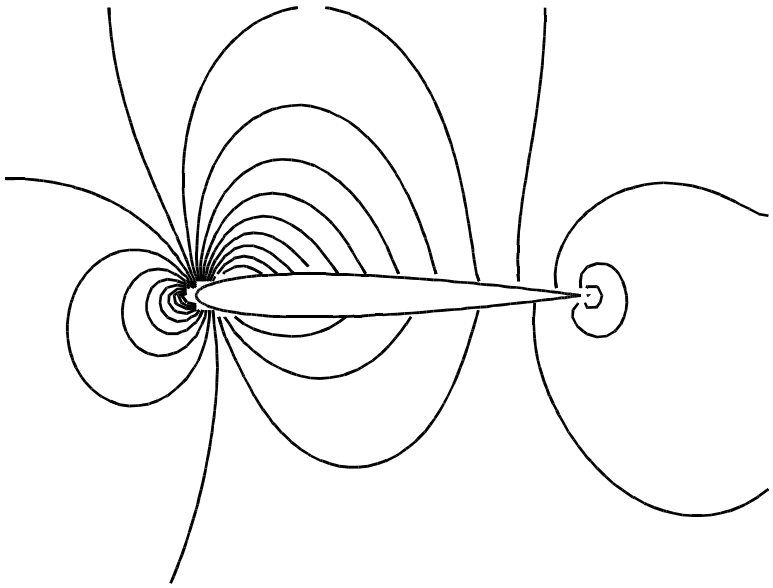


Figure 3.36: Pressure contours for embedded cartesian grid

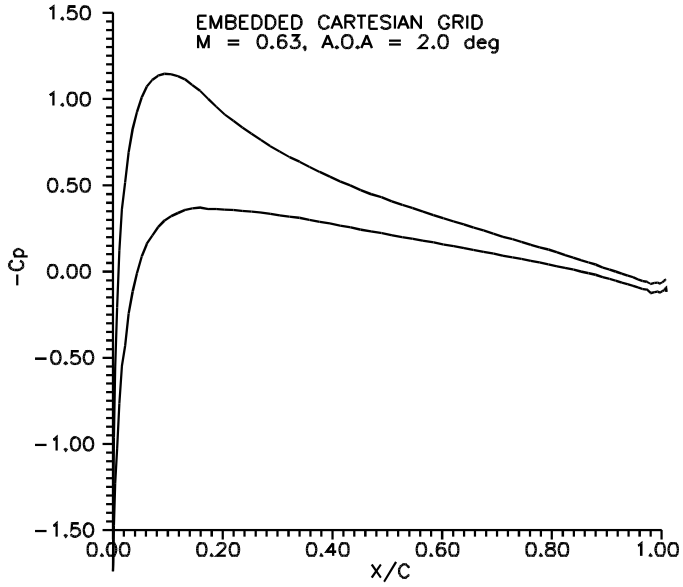


Figure 3.37: C_p Plots : $M = 0.63 \;\; \alpha = 2^o$

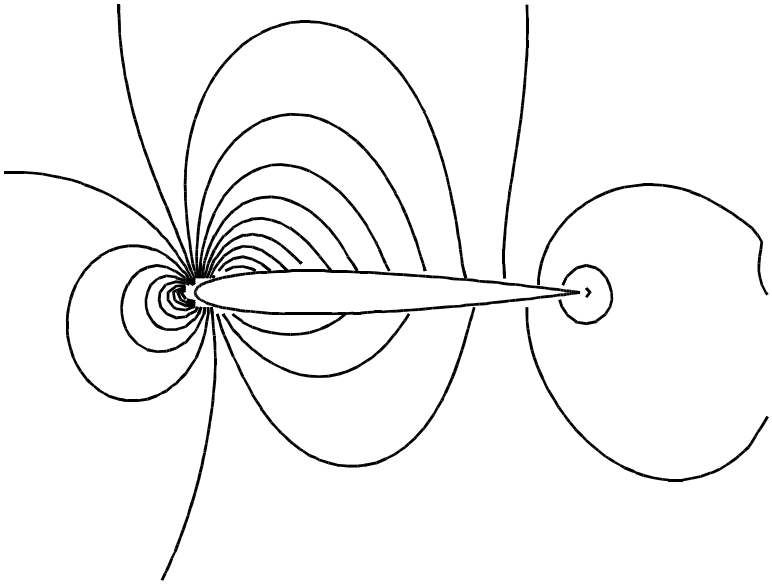


Figure 3.38: Pressure contours for embedded cartesian grid + Polar grid at T.E

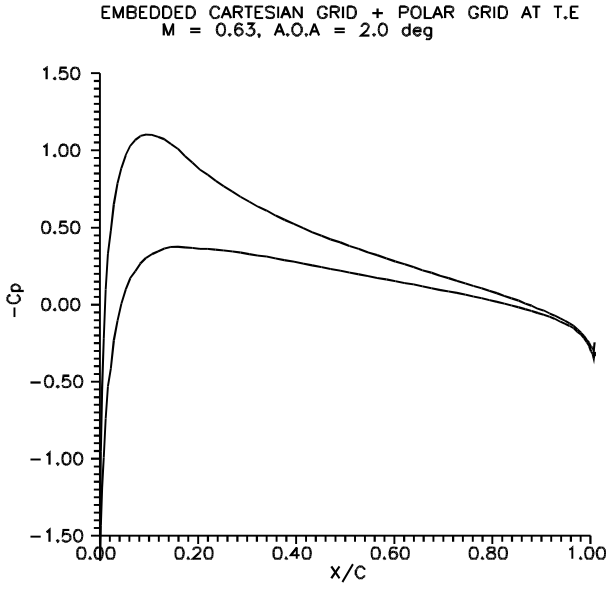


Figure 3.39: C_p Plots : $M = 0.63 \;\; \alpha = 2^o$

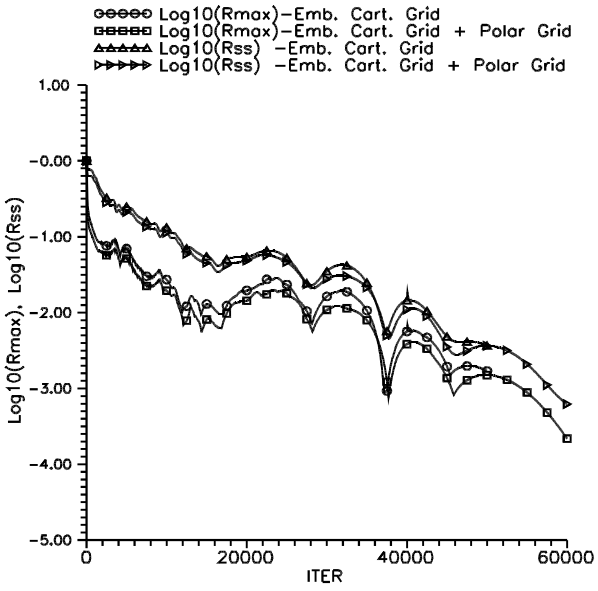


Figure 3.40: Residue Plots - embedded cartesian grid$M = 0.85 \;\; \alpha = 1^o$

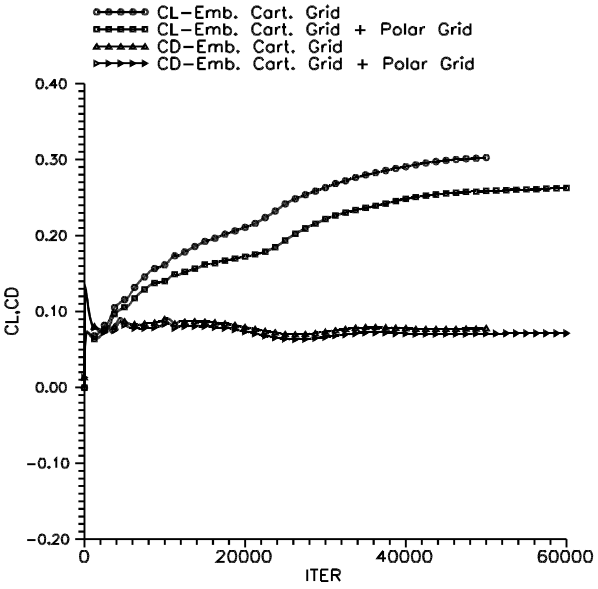


Figure 3.41: C_l C_d convergence - embedded cartesian grid$M = 0.85 \;\; \alpha = 1^o$

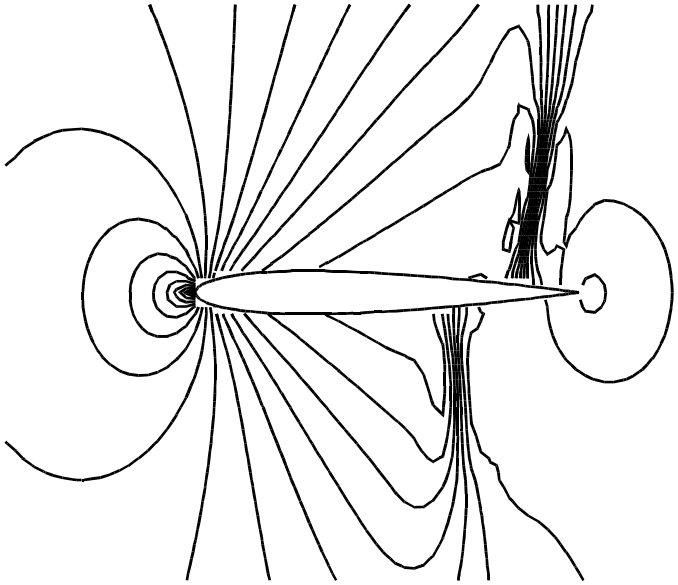


Figure 3.42: Pressure contours for embedded cartesian grid

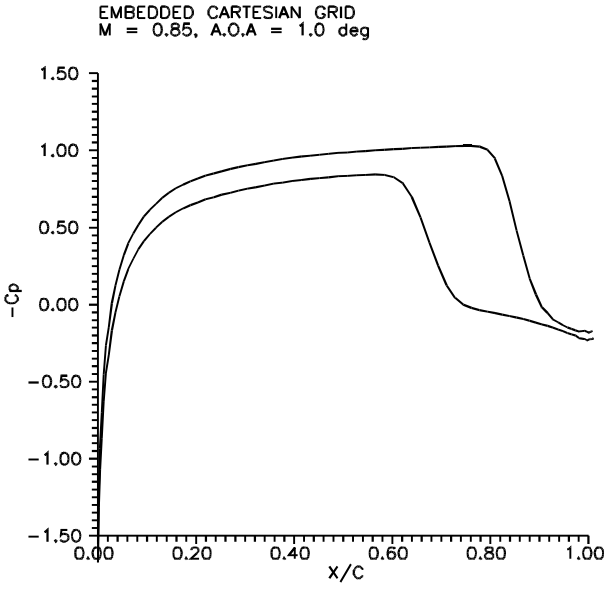


Figure 3.43: C_p Plots : $M = 0.85 \;\; \alpha = 1^o$

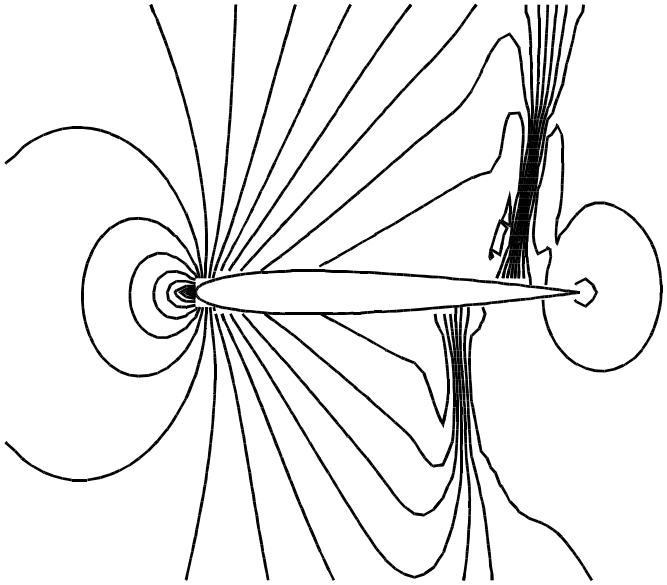


Figure 3.44: Pressure contours for embedded cartesian grid + Polar grid at T.E

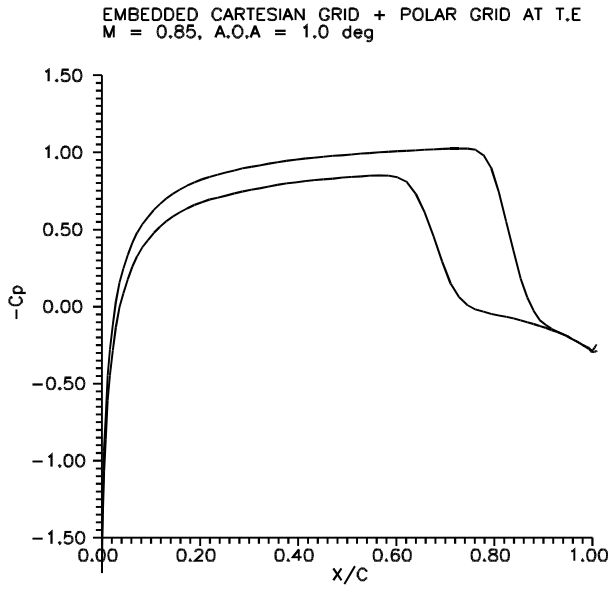


Figure 3.45: C_p Plots : $M = 0.85 \;\; \alpha = 1^o$

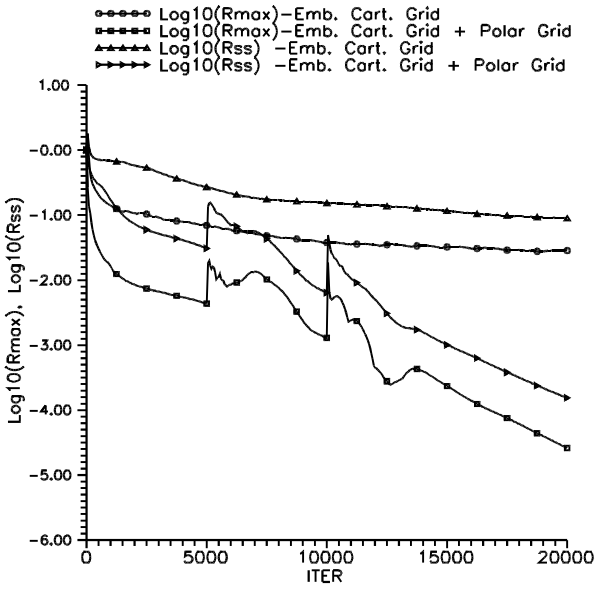


Figure 3.46: Residue Plots - embedded cartesian grid$M = 1.20 \ \alpha = 0^o$

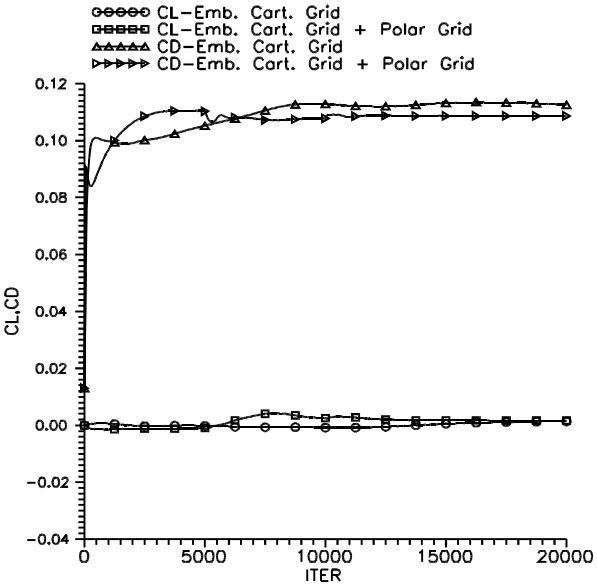


Figure 3.47: C_l C_d convergence - embedded cartesian grid$M = 1.20 \ \alpha = 0^o$

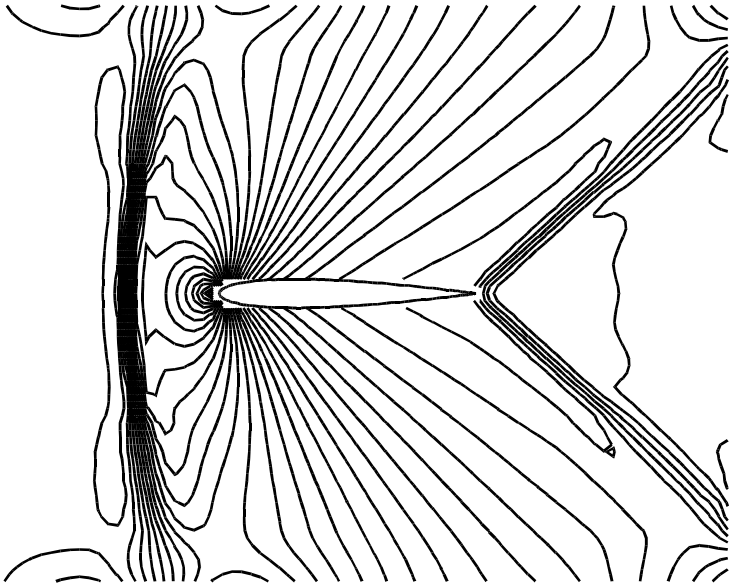


Figure 3.48: Pressure contours for embedded cartesian grid

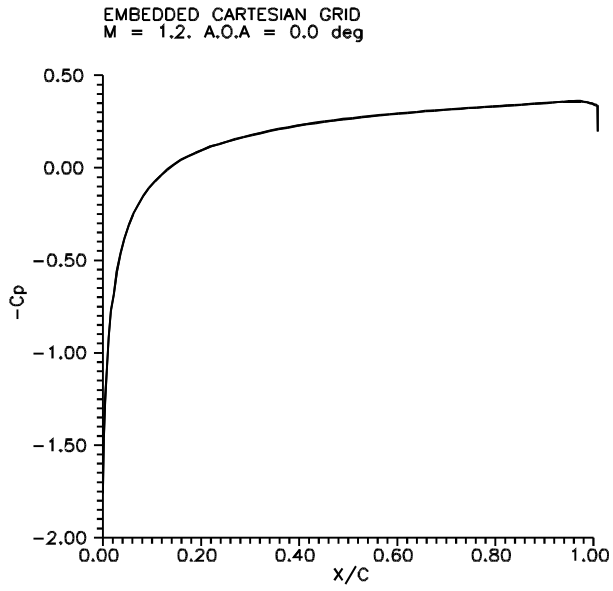


Figure 3.49: C_p Plots : $M = 1.20 \;\; \alpha = 0^o$

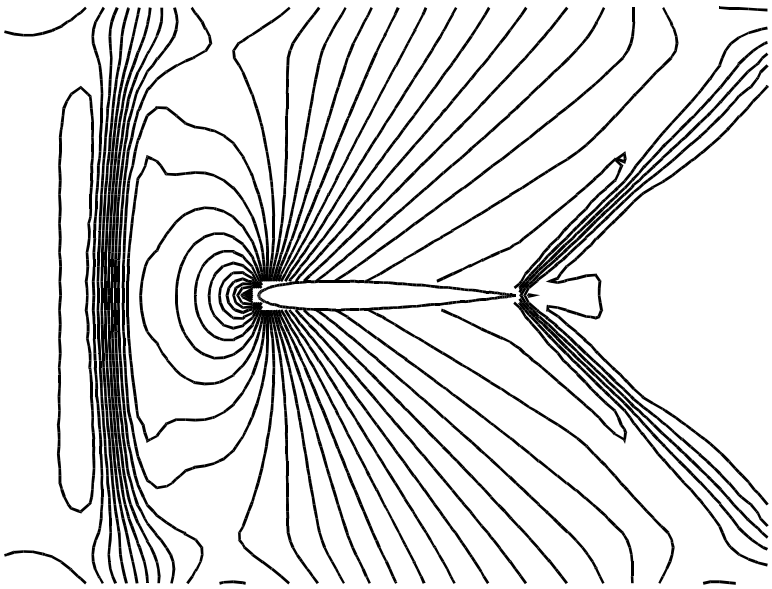


Figure 3.50: Pressure contours for embedded cartesian grid + Polar grid at T.E

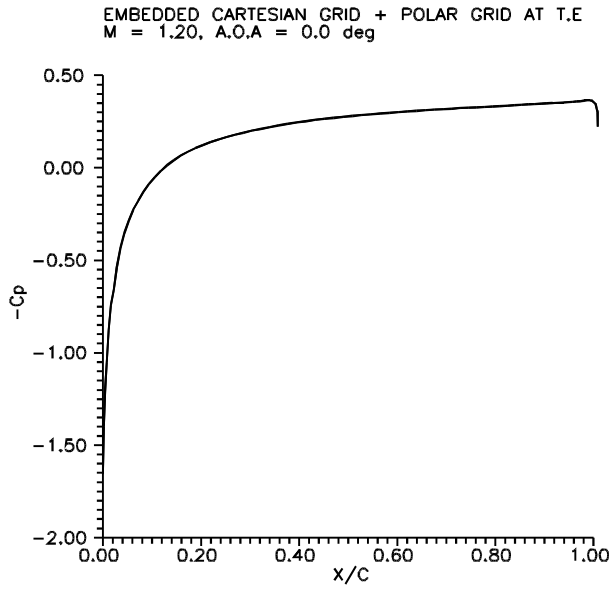


Figure 3.51: C_p Plots : $M = 1.20$ $\alpha = 0^o$

3.5 Results on point distribution from overlapping Structured and Cartesian grid

For this point distribution again we compute for flow past NACA0012 airfoil for subsonic, transonoic and supersonic cases. Figures 3.52 and 3.53 show the residue histories and C_l C_d convergence plots respectively for the subsonic test case. The lift and drag co-efficients settle roughly around 10000 iterations. Figures 3.54 and 3.55 shows pressure distribution and pressure contours around the airfoil for the subsonic case. The results indicate that the predicted pressure distribution is quite smooth. Similarly figures 3.56, 3.57, 3.58 and 3.59 shows the results for the transonic case. Both the upper and lower shocks are well predicted. The C_p distribution near the trailing edge also exhibits a smooth behaviour. Finally the results for the supersonic cases are presented in figures 3.60 to 3.63. All the essential features such as the bow shock in front of the airfoil and the fish tail shock at the trailing edge are satisfactorily predicted. Tables 3.10, 3.11 and 3.12 show the comparison of the computed results for all the test cases with the standard results. It can be observed from these tables that all the computed results compare favourably with the standard test cases.

Grid Type	C_l	C_d
Stru. Grid + Cart. Grid	0.321	0.004
GAMM	0.329-0.336	0.3E-04-0.7E-3

Table 3.10: C_l C_d :overlapping Structured and Cartesian grid: $M = 0.63$ $\alpha = 2.0^o$

Grid Type	C_l	C_d
Stru. Grid + Cart. Grid	0.305	0.057
AGARD	0.330-0.389	0.0464-0.0590

Table 3.11: C_l C_d :overlapping Structured and Cartesian grid:$M = 0.85$ $\alpha = 1.0^o$

Grid Type	C_l	C_d
Stru. Grid + Cart. Grid	-0.0005	.099
AGARD	should be 0	0.0946-0.096

Table 3.12: C_l C_d :overlapping Structured and Cartesian grid:$M = 1.20$ $\alpha = 0.0^o$

The computed results for this point distribution consistently gives better results compared to the point distribution from embedded cartesian grids. This is because the with the present point distribution we have a better resolution of the field near the body. So far we have considered different types of meshes for a single element

case. In the next section we consider the NACA0012 airfoil biplane configuration. For this case we have point distribution obtained from two overlapping structured O-grids with a background Cartesian grid. Figure 3.7 shows a view of the point distribution for this case. The two body conforming grids adequately resolves the field near the airfoils. They extend upto about 1.5 chord distance from the body. Beyond these two overlapping grids we have the background Cartesian grid.

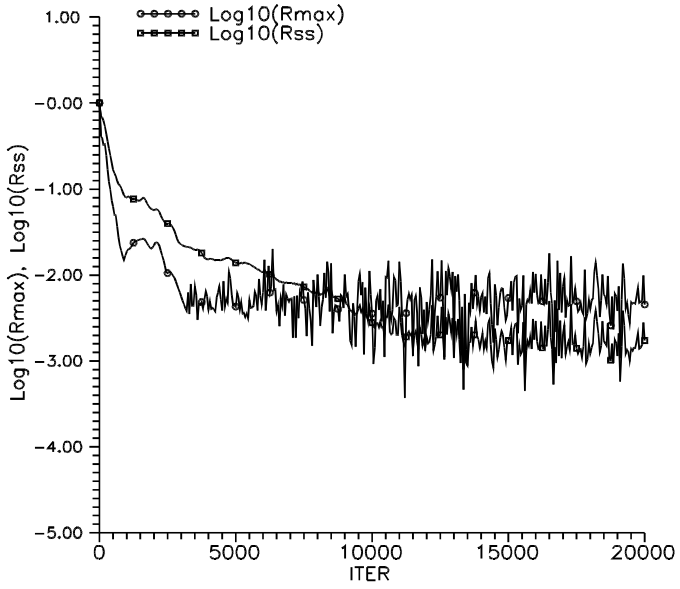


Figure 3.52: Residue Plots : overlapping structured-O grid and cartesian grid $M = 0.63$ $\alpha = 2^o$

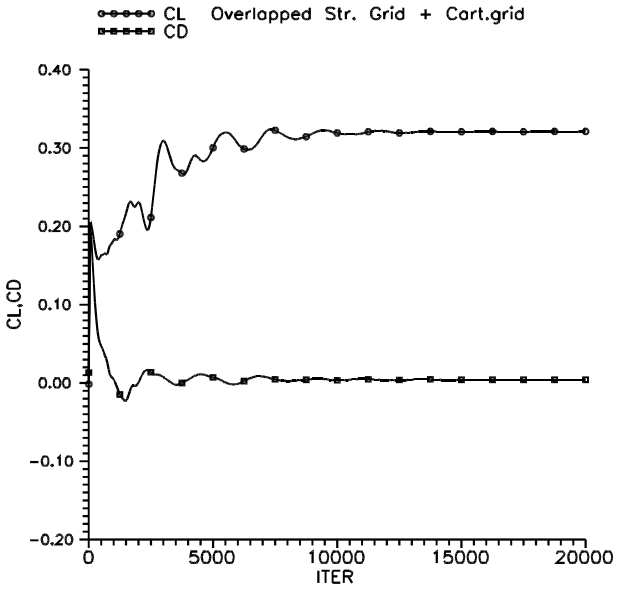


Figure 3.53: C_l C_d: convergence : $M = 0.63$ $\alpha = 2^o$

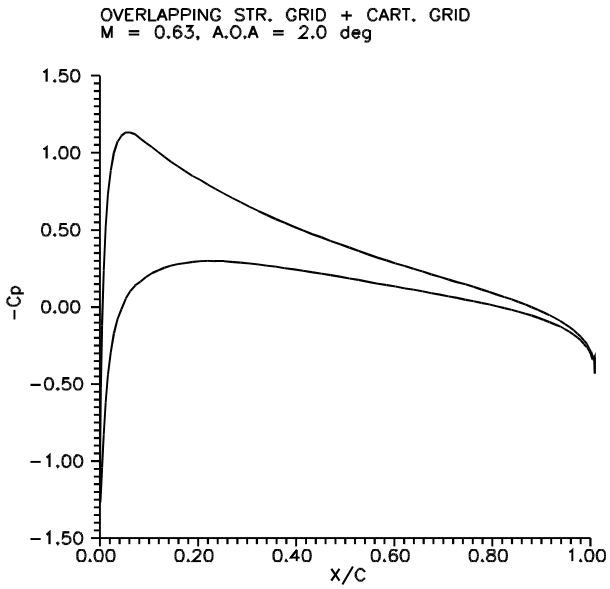


Figure 3.54: Cp Plots : $M = 0.63 \;\; \alpha = 2^o$

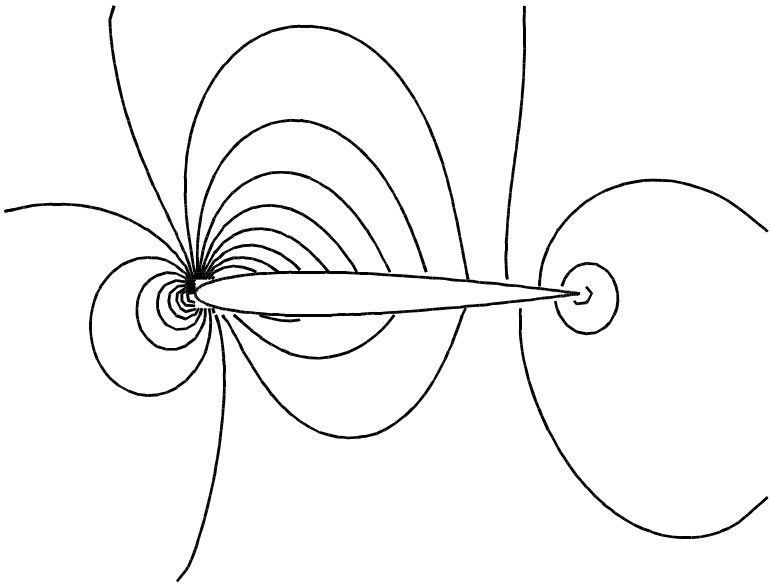


Figure 3.55: Pressure contours $M = 0.63 \;\; \alpha = 2^o$

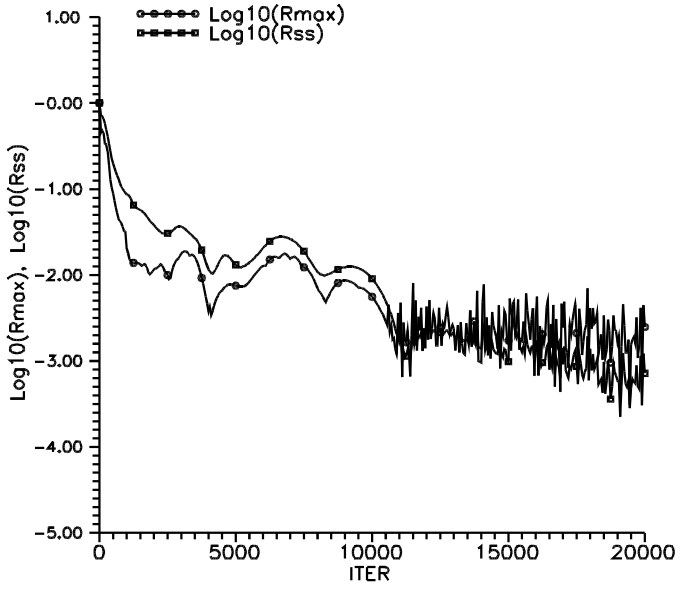


Figure 3.56: Residue Plots : overlapping structured-O grid and cartesian grid $M = 0.85$ $\alpha = 1^o$

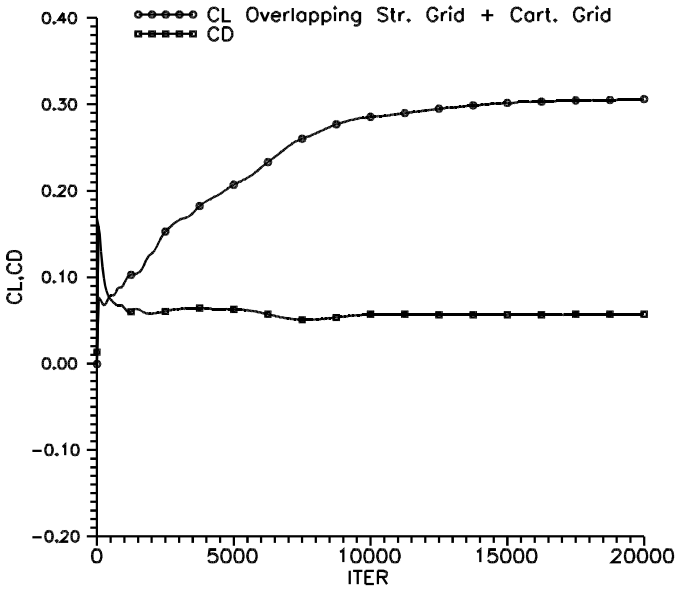


Figure 3.57: C_l C_d convergence : $M = 0.85$ $\alpha = 1^o$

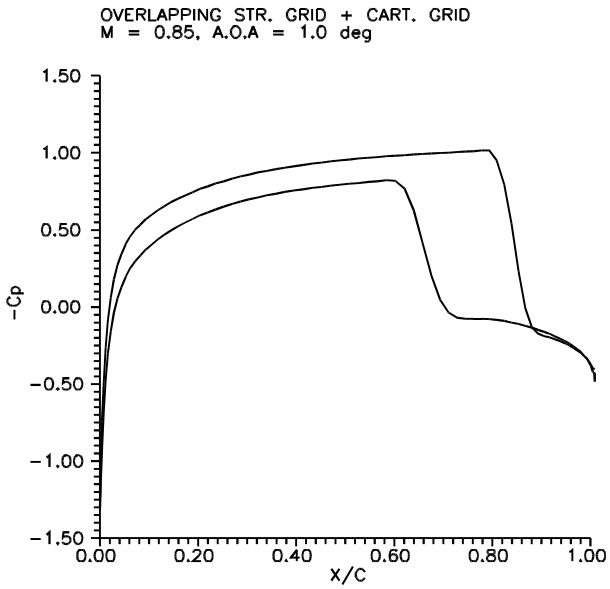


Figure 3.58: Cp Plots : $M = 0.85 \;\; \alpha = 1^o$

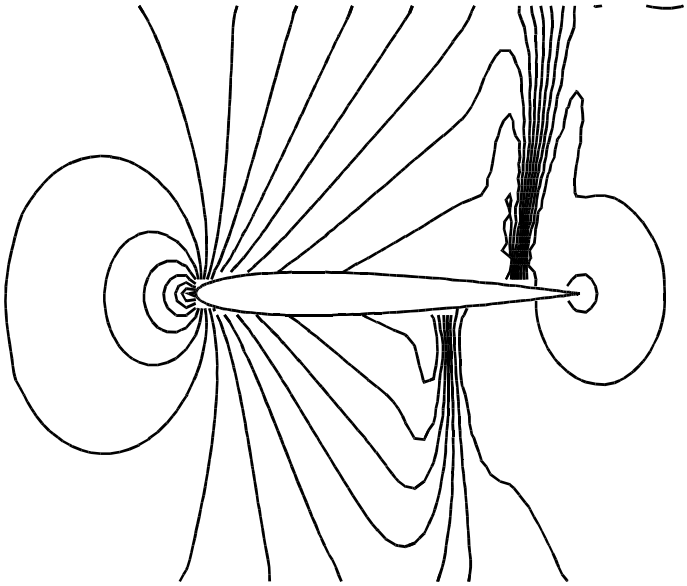


Figure 3.59: Pressure contours $M = 0.85 \;\; \alpha = 1^o$

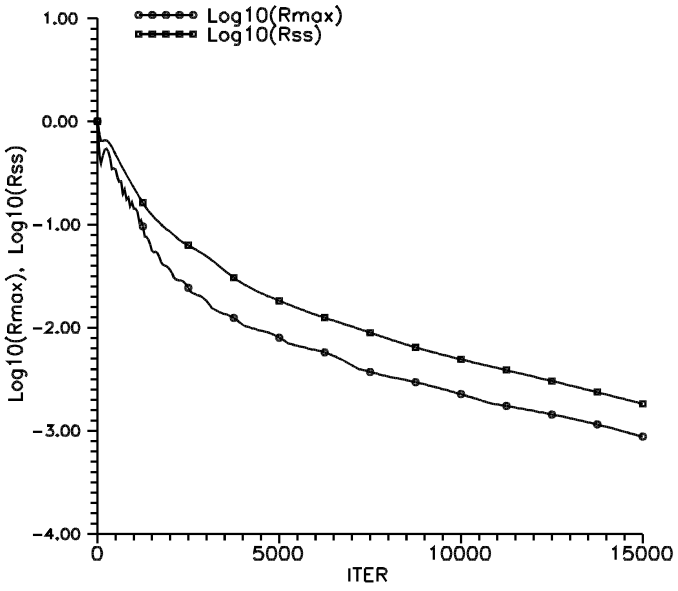


Figure 3.60: Residue Plots: $M = 1.20 \;\; \alpha = 0^o$

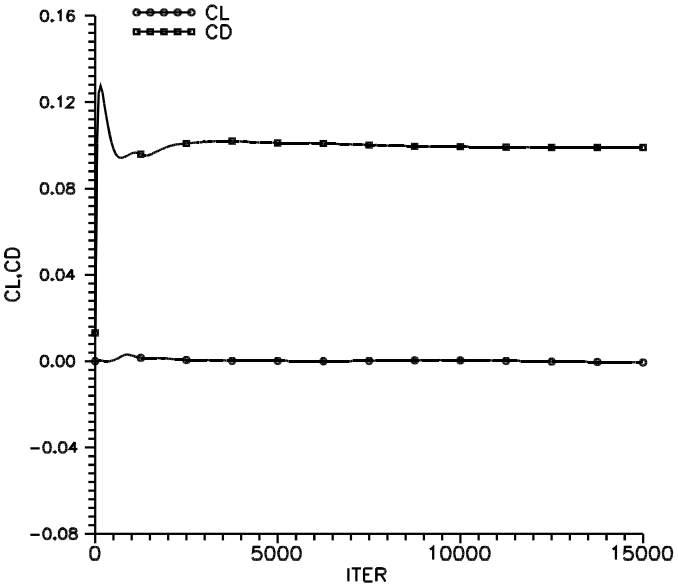


Figure 3.61: C_l C_d convergence : $M = 1.20 \;\; \alpha = 0^o$

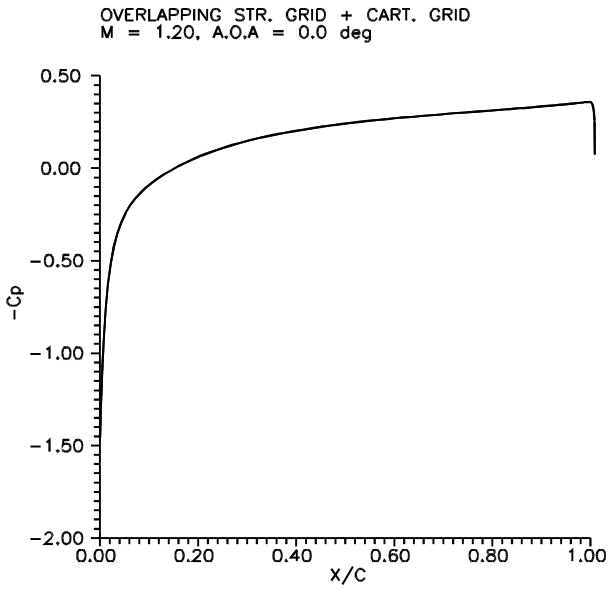


Figure 3.62: Cp plots :$M = 1.20 \;\; \alpha = 0^o$

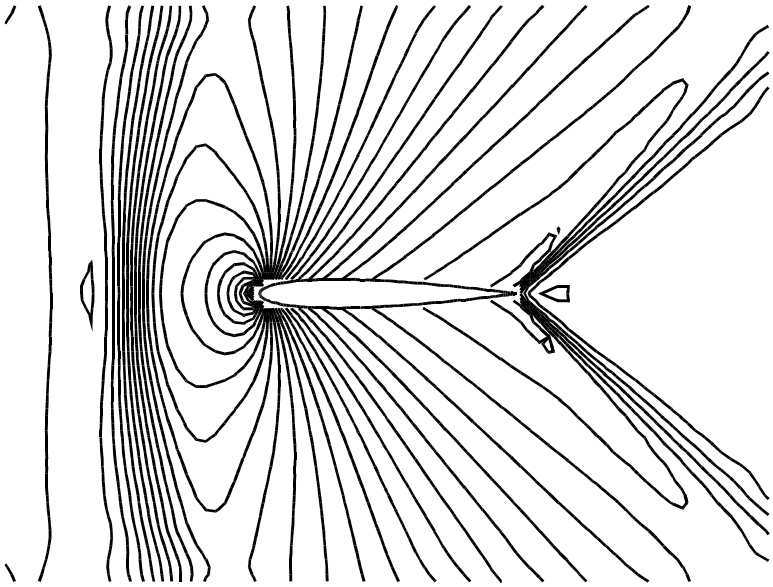


Figure 3.63: Pressure contours : $M = 1.20 \;\; \alpha = 0^o$

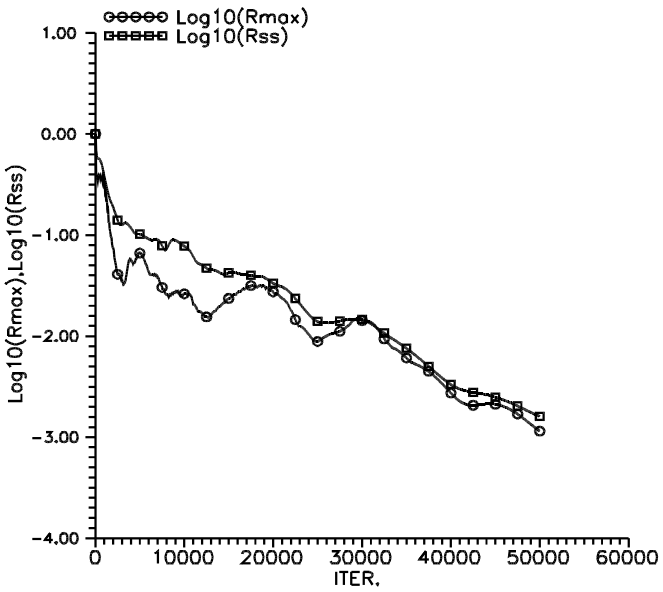


Figure 3.64: Residue Plots - NACA0012 Biplane

3.6 Results on point distribution from Overlapping Structured Grids with Background Cartesian mesh

For this grid we have considered transonic flow past the NACA 0012 airfoil biplane configuration. Figure 3.64 shows the residue plots for this case. Figure 3.65 shows the computed pressure contours. Figure 3.66 and 3.67 shows the pressure coefficient distribution over the upper and the lower airoils. It can be observed from these figures all the essential features of this case are very well predicted. This further supports the view that using independent body fitted mesh around each element and a background cartesian mesh away from the interior region would be a possible choice for solving multibody configurations.

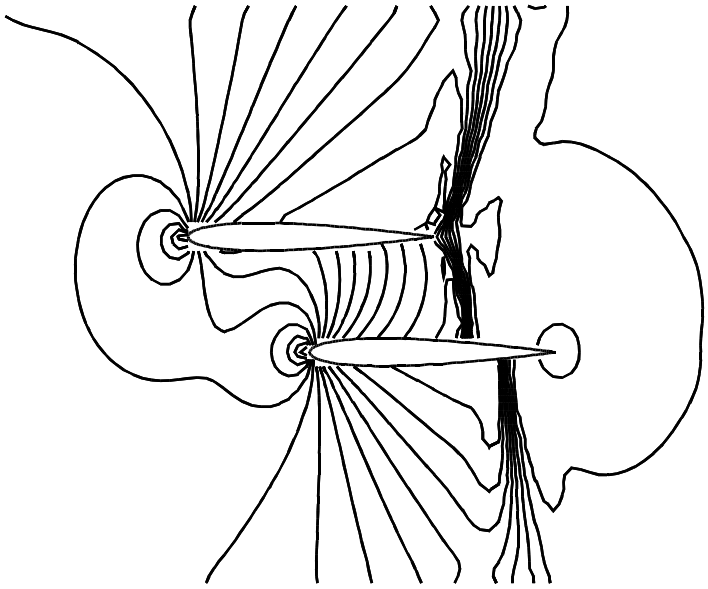


Figure 3.65: Pressure contours : Biplane configuration

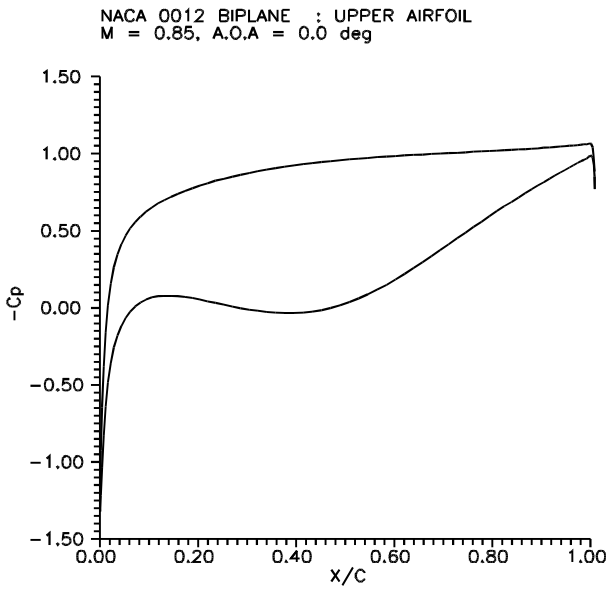


Figure 3.66: Cp Plots Upper Airfoil : $M = 0.85 \;\; \alpha = 0^o$

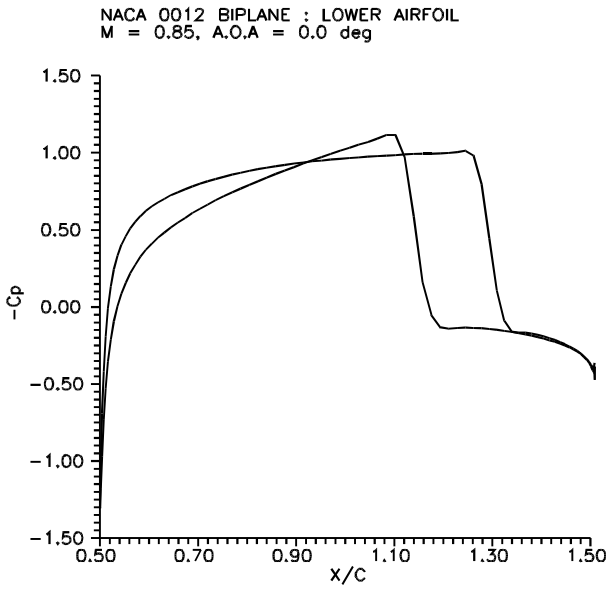


Figure 3.67: Cp Plots Lower Airfoil : $M = 0.85 \;\; \alpha = 0^o$

3.7 Conclusions - 2D Computations

The LSKUM solver has been shown to work on a variety of point distributions, thus showing its grid free nature. In this chapter we have amply demonstrated this capabiltiy of LSKUM solver to work on a variety of distribution of points obtained from structured, unstructured, Cartesian grids as well as chimera grids. In all these applications we also found that the quadtree pre-processor is the most important tool in the generation of connectivity data. This preprocessor enables the LSKUM solver to operate on any distribution of points with the associated connectivity data. There are distinct advantages of the present approach while using points from overlapped grids. Conventional methods[6, 82, 32] using overlapped meshes have to keep track of different boundaries of each grid, that is, inner boundary of overlap region, outer boundary of overlap region. Suitable interpolation schemes are required for solution transfer across different boundaries of the overlap region. The present approach does not require division of the computational domain into overlap domain and domain with overlapping grids. Finite volume methods operating on chimera meshes require seperate solution procedure for overlap domain. The LSKUM solver is the same throught the entire domain. The solver simply operates on the total set of points obtained from all the overlapping grids. In fact in the overlapped regions conventional solvers introduce numerical errors due to the interpolation procedures adopted. But in the present method we have more nodes to resolve the flow in the overlapped region.

In the present work we have developed and used the new Kinetic theory based Outer boundary condition(KOBC). This has a distinct advantage that, the treatment of the boundary condition is the same for all types of flow conditions at the boundary.

Having got very encouraging results for 2-D, we proceed to apply LSKUM to 3-D problems. The formulation we have already discussed in the first chapter.

Chapter 4

Results for 3-D LSKUM

In the previous chapter we have demonstrated the grid free capability of the LSKUM solver for a wide variety of 2-D problems. This success prompts us to extend the method to 3-D problems. Ideally such grid free capability of the solver would be highly desirable for 3-D practical applications. Gridding for complex geometrical configurations has always been a problem. One popular approach is to use multiblock grid, that is, first divide the computational domain into several topologically simpler domains called blocks. Then 3D grid generator can be employed to grid the block. Yet another possibility is to use tetrahedral mesh(so called unstructured mesh) in the computational domain. This task requires a very sophisticated unstructured mesh generator. We are following a different approach. The idea in this approach is to generate a cloud of points around the complex configurations using a simple, cheap and easy to use grid generator for every component into which the complex geometry can be divided. We then get a multiple overlapping chimera mesh. We have tested this approach for solving several 2D problems and the results obtained are very encouraging. Before extending LSKUM preprocessor combination to complex 3D configurations it is neccessary first to establish 3D LSKUM Euler solver for relatively simpler 3D geometries. Towards this goal we have made a beginning in exploiting the power of LSKUM solver for 3-D problems. In the present work we have developed a 3-D LSKUM solver and applied to some standard test cases. In the next section we give the details of the problems chosen for 3-D LSKUM.

4.1 Test Cases for 3-D LSKUM

We have considered the following test cases for the purpose of validating the 3-D LSKUM code.

1. Simulation of an intense blast wave
2. Supersonic flow past Hemisphere
3. Transonic flow past ONERA-M6 wing

4. Hypersonic flow past a generic flight vehicle

The first test case consists of points from a simple uniform cartesian grid. Even though this test case consists of simple grid, the flow conditions are very extreme. Also this case does not involve any solid boundary conditions. In this case we basically wish to test the ability of the code to compute flow even under extreme flow conditions, and to establish the correctness of the various formulae for the split fluxes and its derivatives used in the code. For this blast wave problem as well as for the all the other 3D test cases we have used for the first time our new 3D Kinetic Outer Boundary Condition(KOBC). We thus also wish to establish the working of this new boundary condition for 3D problems. In the second test case we consider the supersonic flow past a hemisphere. This is a simple example of 3D flow. Yet with this test case we establish the correctness of the solid wall boundary conditions which was not present in the first case. After having fully tested for the interior scheme as well as the boundary scheme(inclusive of solid wall and the far field) we then apply the 3D LSKUM solver for the well known test case of transonic flow past ONERA-M6 wing. Finally after having tested the 3D LSKUM solver for these test cases, we compute the flow past a fairly complex geometry. This final test case is a generic flight vehicle consisting of bluntcone-cylinder-flare with lifting surfaces. Computed results are compared with available experimental data.

4.2 Simulation of an intense blast wave using rectangular grid

For this test case we have chosen points from an uniform Cartesian grid. The purpose of this calculation is to test the robustness of the method for computing flow fields involving very high temperature and pressure gradients. The propagation of a blast wave in air is spherically symmetric. Therefore it is natural for one to choose points from a spherical grid for computations. However since our method is grid independent, we wish to test the ability of the method to predict the spherical waves even when we use points from an uniform Cartesian grid. The test conditions chosen for the simulation of the blast wave corresponds to the first atomic explosion in New Mexico(1947). The results of the present computations are validated against the experimental data and the similarity solution reported by G.I.Taylor in 1950[73, 74].

We give here some relevent data corresponding to this explosion which has been used as input for the present computation. The total energy released E has been estimated[74] to be equivalent to $16,800$ tons of T.N.T or approximately 7.14×10^{13} Joules. The grid size chosen for this problem is $81 \times 81 \times 81$ and the computational domain is of size $81m \times 81m \times 81m$. To start with the computations at time $t = 0$, this amount of energy is assumed to be concentrated in a cubical core of size $10m \times 10m \times 10m$. The temperature in the core to start with is estimated as follows.

$$\text{Mass in the core } m \quad = \quad \rho \text{ Volume}$$

$$= \quad 1.228 \times 10^3 \text{ kg}$$

$$\text{Energy } E \quad = \quad mC_vT_{core} \quad (C_v = 717.7\text{J/Kg})$$

with these simplified assumptions one gets the temperature in the core to begin with as $T_{core} = 8.1 \times 10^7$k. The pressure in the core is then assumed to be $p_{core} = \rho * R * T_{core}$. With this as the temperature and pressure in the core and assuming $T_{ambient} = 298$k and atmospheric pressure throught the domain(except in the core) we start the computations.

Reference[74] gives the radius of the blast wave, after the initiation of the blast wave,from 0.1ms(milli second) to 62ms, with the radius of the blast wave growing from 11.1m to 185m. However due to the restriction of the outer boundary extent and the maximum number of the grid points we could have, computations have been done upto time instant of 0.80ms. All the computations for this case are done for first order with uniform weights.

Figures 4.1 to 4.5 show the pressure contours for various time instants in the central plane of the cube. It can be seen that that the spherical symmetry of the blast wave is prefectly predicted given the fact that we have used an uniform Cartesian grid. This we feel that, it is because of the grid free nature as well as the robustness of the method we have been able to predict these spherical blast waves with extremely high temperatures and pressures in the flow field. It has been shown by G.I.Taylor[74] that except for the very initial stages of the blast, the radius of the blast wave R for any time instant t after the initiation of the blast wave, the product $R^{\frac{5}{2}}t^{-1}$ is constant. Figure 4.6 shows a comparitive plot for the experiment and the predicted values of $\frac{5}{2}\log(R)$ against $\log(t)$,with the line corresponding to $R^{\frac{5}{2}}t^{-1} =$ constant drawn in the figure. It can be seen that an excellent agreement is obtained, except for the very initial stage of the blast, (ie at time = 0.1ms) where the conditions are expected to be highly transient. Even though we obtain a very good comparison, one would be surprised, as to how one can get these for the assumption of $\gamma = 1.4$ (the ratio of specific heats) for the extremely high temperatures. Also the similarity solution obtained by G.I.Taylor makes a similar assumption. Here we merely quote the probable reasons argued out by him for getting meaningful results even for the assumption of $\gamma = constant$. He says([74] page number 177).**"The ball of fire did therefore expand very closely with the theoretical prediction made more than four years before the explosion took place. This is surprising, because in those calculations it was assumed that air behaves as though γ, the ratio of the specific heats, is constant at all temperatures, an assumption which is certainly not true. At room temperatures $\gamma = 1.4$ in air, but at high temperatures γ is reduced owing to the absorption of energy in the form of vibrations which increases C_v. At very high temperatures γ may be increased owing to dissociation. On the other hand, the existance of very intense radiation from the centre and absorption in the outer regions may be expected to raise the apparent value of γ. The fact that the observed value of R^5t^{-2} is so nearly constant through the whole range of**

radii covered by the photographs of the ball of fire suggests that these effects may neutralize one another, leaving the whole system to behave as though γ has been an effective value identical with that which it has when none of them are important, namely, 1.40." Figures 4.7, 4.8 and 4.9 show the plot of predicted density,pressure and temperature respectively along the radial direction for different time instants. It can seen that the maximum pressure location corresponds to the shock wave location. The maximum pressure also drops with time, which is expected. Table 4.1 shows a comparison of the experimental and the predicted values of the Radius of the shock wave.(Inferred from the plot of pressure versus radial distance)

T(msec)	R(Experiment)	R(computation)
0.10	11.1	14.1
0.24	19.9	19.7
0.38	25.4	25.0
0.52	28.8	28.3
0.66	31.9	31.1
0.80	34.2	33.9

Table 4.1: Comparison of the Radius of the Blast wave

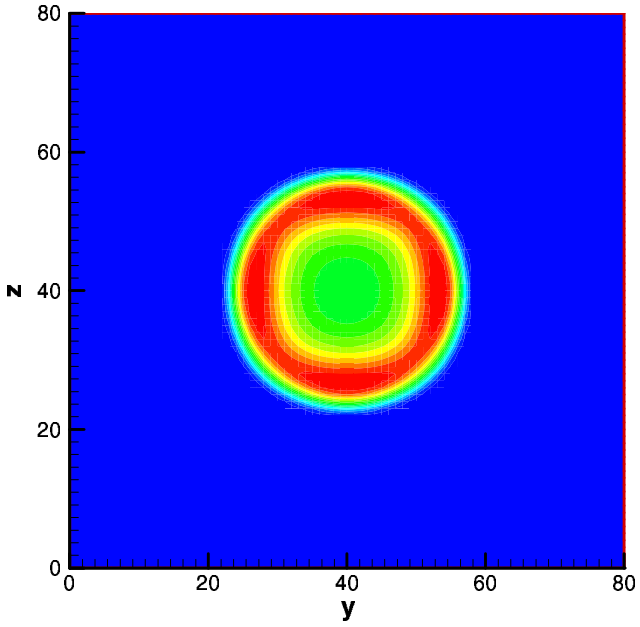


Figure 4.1: Pressure Contours at 0.1 ms

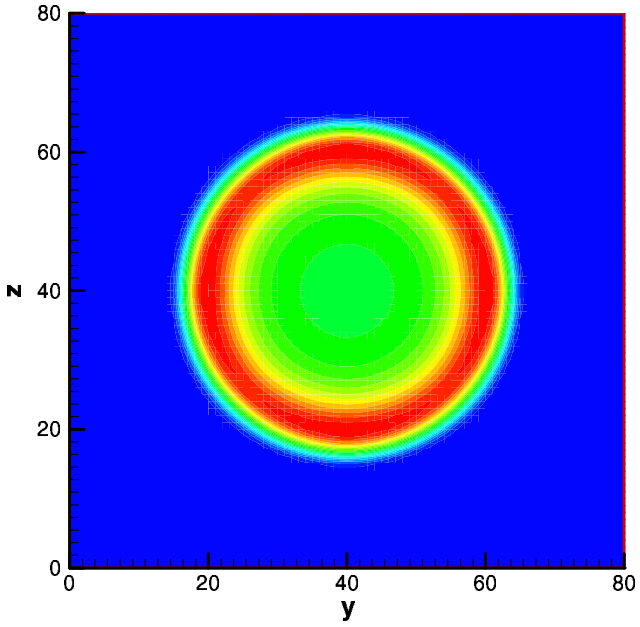


Figure 4.2: Pressure Contours at 0.24 ms

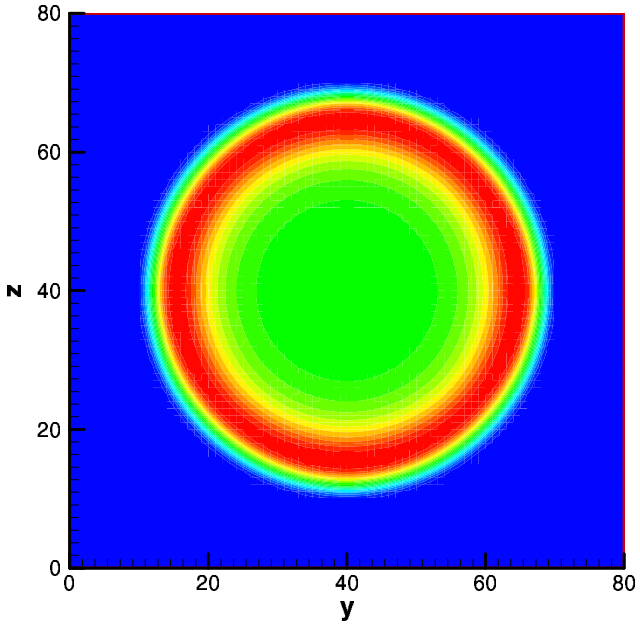


Figure 4.3: Pressure Contours at 0.38 ms

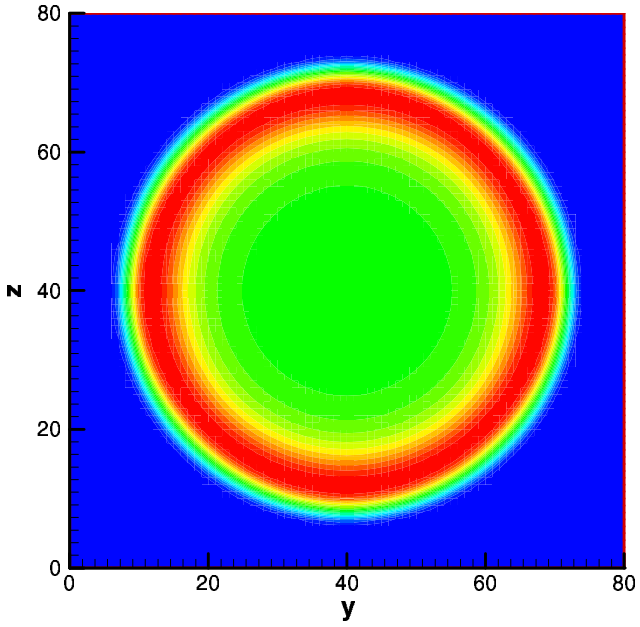


Figure 4.4: Pressure Contours at 0.52 ms

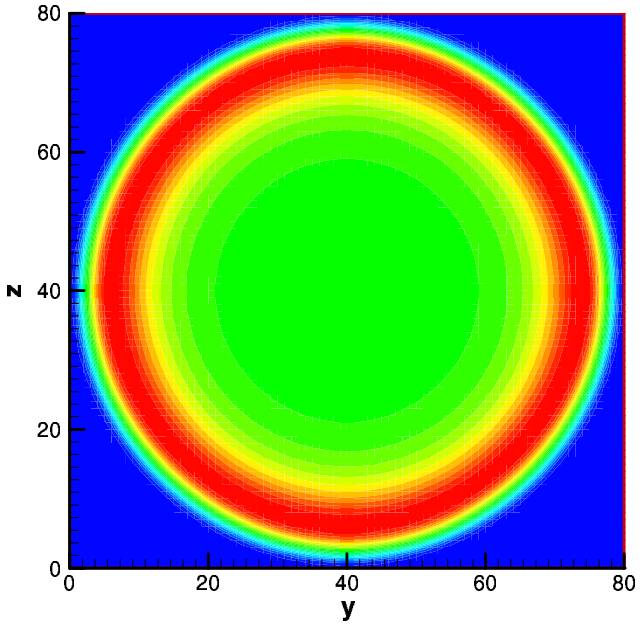


Figure 4.5: Pressure Contours at 0.80 ms

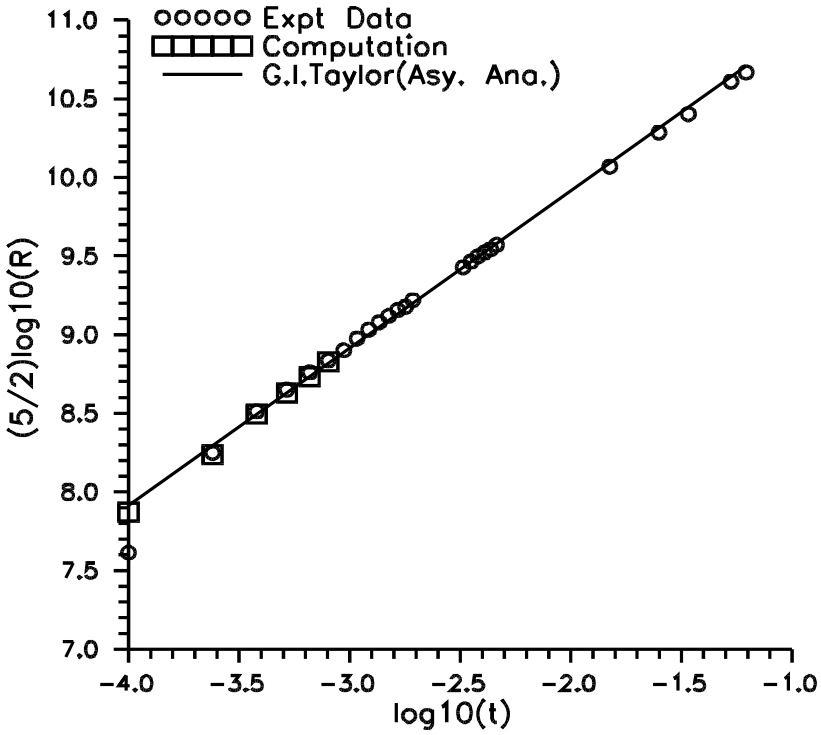


Figure 4.6: Comaprison of the computed radius of the Blast wave front

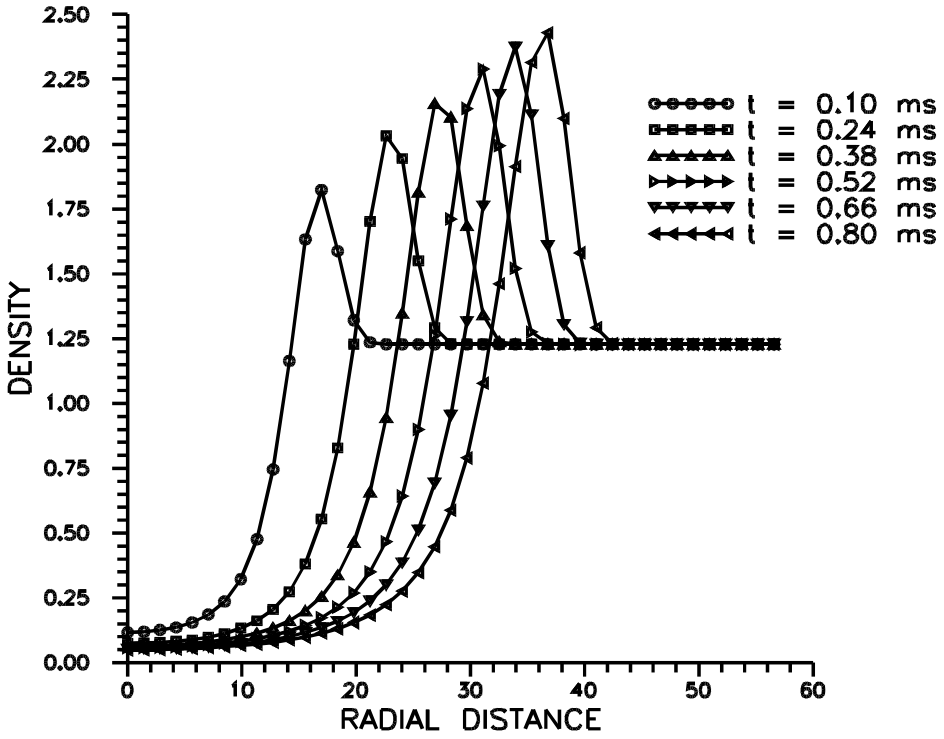


Figure 4.7: Radial Variation of Density at different time instants

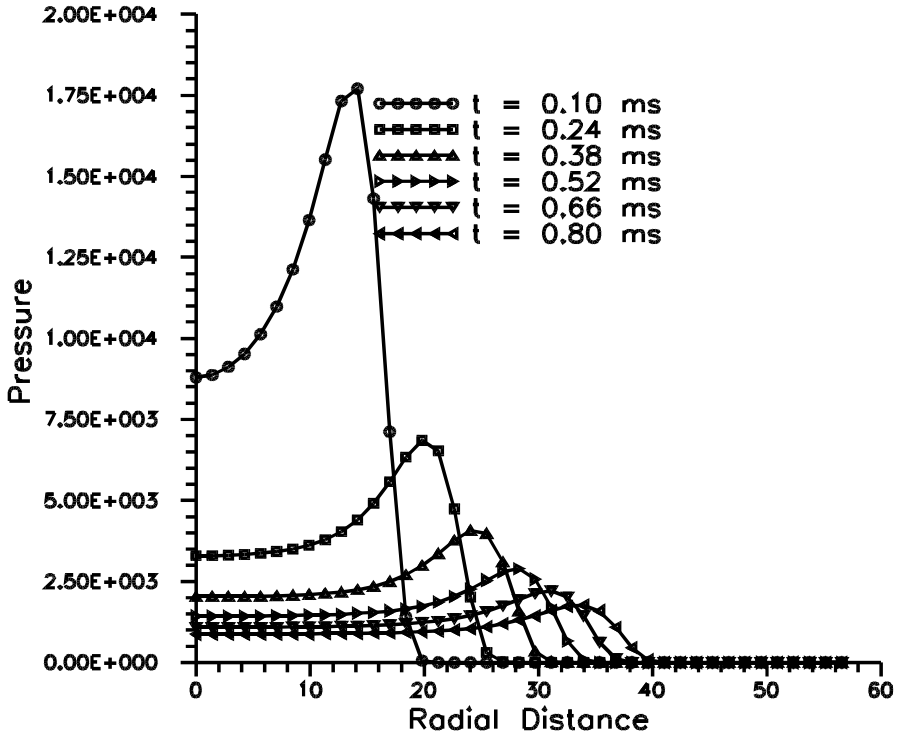


Figure 4.8: Radial Variation of Pressure at different time instants

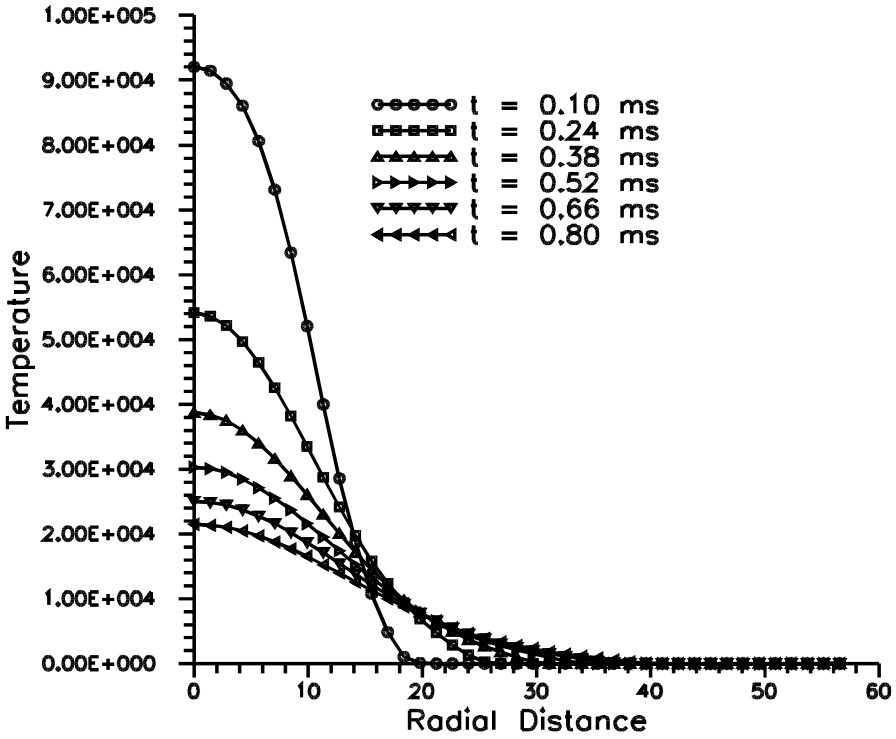


Figure 4.9: Radial Variation of Temperature at different time instants

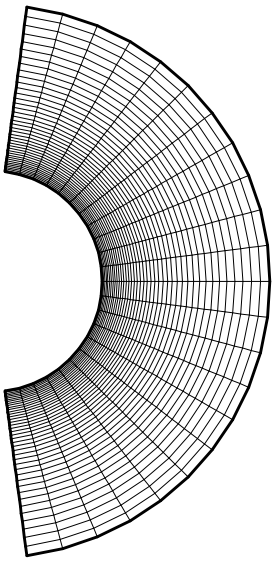


Figure 4.10: A view of the hemispherical grid in the center plane

4.3 Supersonic flow past Hemisphere

For this test case, free stream Mach number is $M_\infty = 3.0$, outer boundary is at a distance 2.5 times the radius of the hemisphere. Points from a structured grid of size $25 \times 25 \times 41$ is used.

Figure 4.10 shows a plot of the grid. Figure 4.11shows the Mach contours for first order comptations. It can be seen from these contours, the first order calculation itself captures the flow features satisfactorily. Figure 4.12 shows a plot of the total pressure along the centreline. Also the exact position of the shock is plotted in the figure based on Billing's emperical relation[2]. It can seen that the position of the shock is satisfactorily predicted. Also the second order solution shows further improvement in the predicted shock stand off distance. Figure 4.13 shows the plot of Mach number along the centreline.

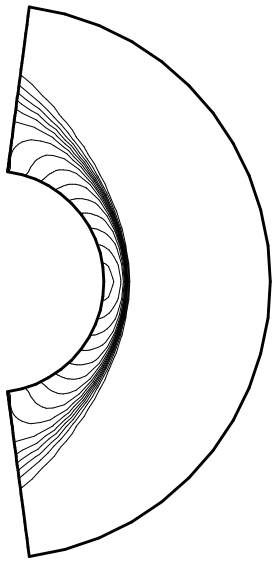


Figure 4.11: Mach contours - Supersonic flow past hemisphere M = 3.0

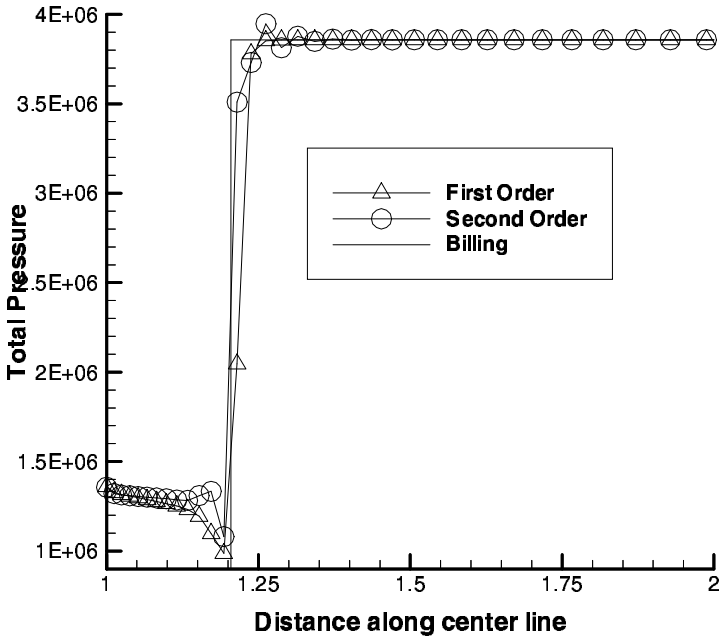


Figure 4.12: Total Pressure variation along center line

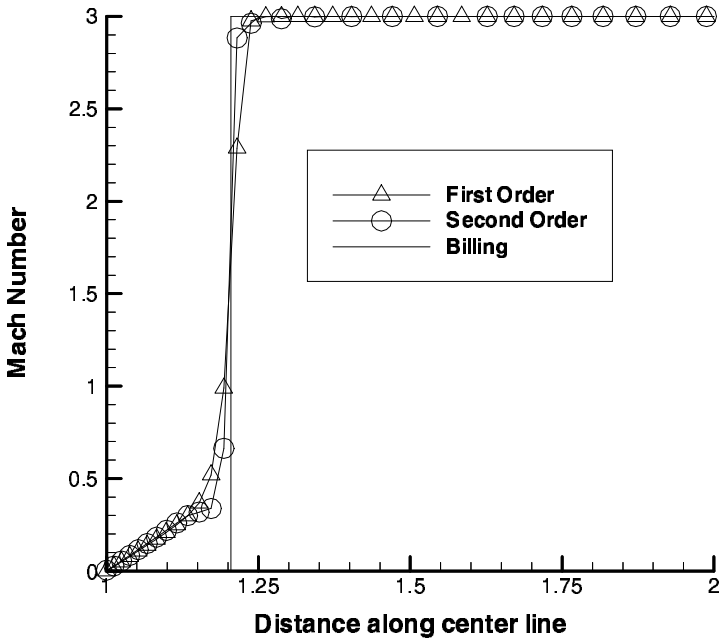


Figure 4.13: Mach Number variation along center line

4.4 Transonic flow past ONERA-M6 wing

In the first test case we considered a flow with extremely high temperature and pressure gradients. However this test case did not contain any solid boundaries. Therefore in order to validate the code for boundary condition treatment we next considered the supersonic flow past hemisphere. For this case even though solid boundary is present the geometry is fairly simple. In order to validate the test case for a fairly complex geometry we have chosen the test case of transonic flow past ONERA-M6[4] wing. This is a standard test case usually chosen to evaluate any new method. For this test case we have chosen points from an O-H type of grid. The grid size used in the present computation is $161 \times 51 \times 31$. We have 161 points in the chordwise sections, 51 in the spanwise direction(with 35 sections on the wing) and 31 points in the direction normal to the wing surface. Figure 4.14 shows a partial view of the grid. This grid was generated using the structured grid generator developed by Mathur and Chakrabartty[57]. The computations have been done using single step Euler for time discretisation and second order accurate discretisation for space derivatives. We have typically about 36 points in connectivity for each node. Also we have used a weighting factor $wt = \frac{1}{d^4}$ for weighted least squares approximation of the derivatives. The mach number of the freestream is $M = 0.84$ and $\alpha = 3.06^o$.

Figure 4.15 shows the residue plot. The solution has fairly converged as can be observed from the figure. Figures 4.16 and 4.17 show the predicted pressure contours. One can see that the major flow feature, that is the double shock on the upper surface with both merging towards the tip side has been captured satisfactorily. Figures 4.18 to 4.23 show a comparison of the computed pressure distribution with the experiment at various span locations. One can see that a fairly good comparison has been obtained.

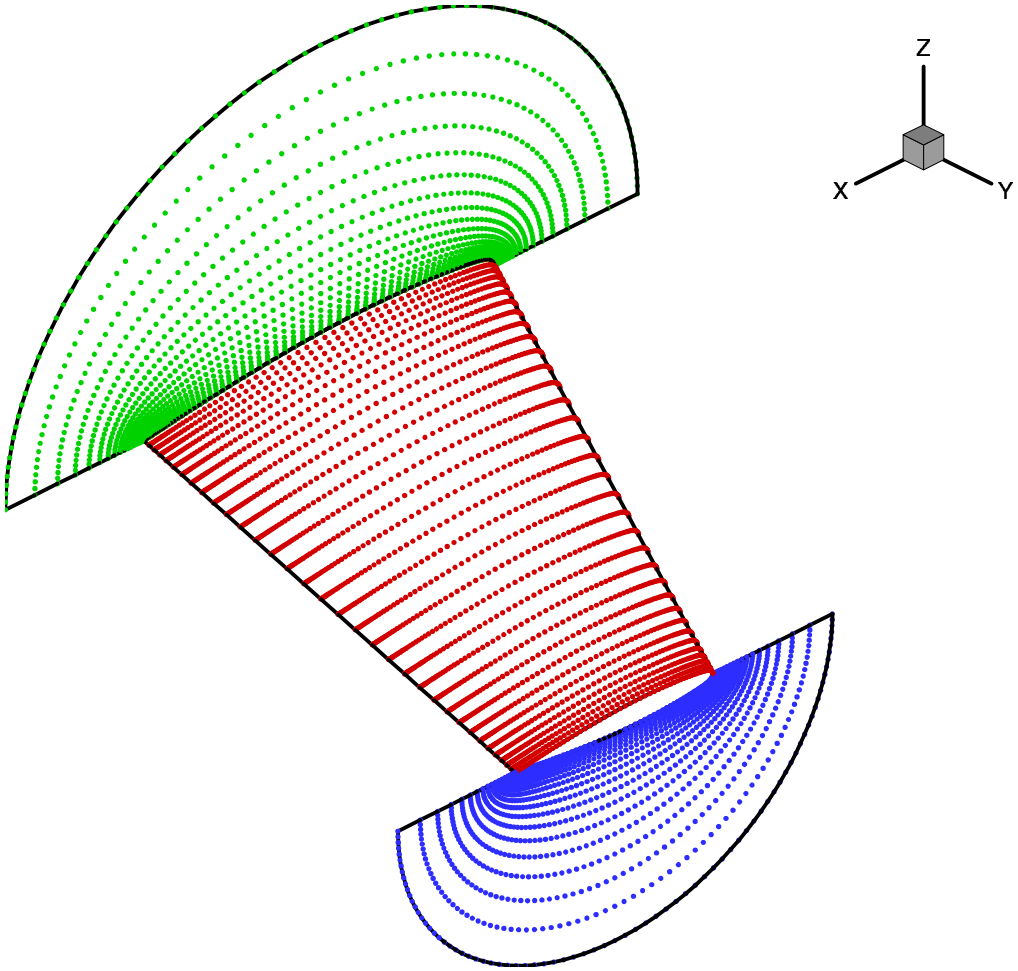


Figure 4.14: A Partial view of the O-H grid over ONERA-M6 wing

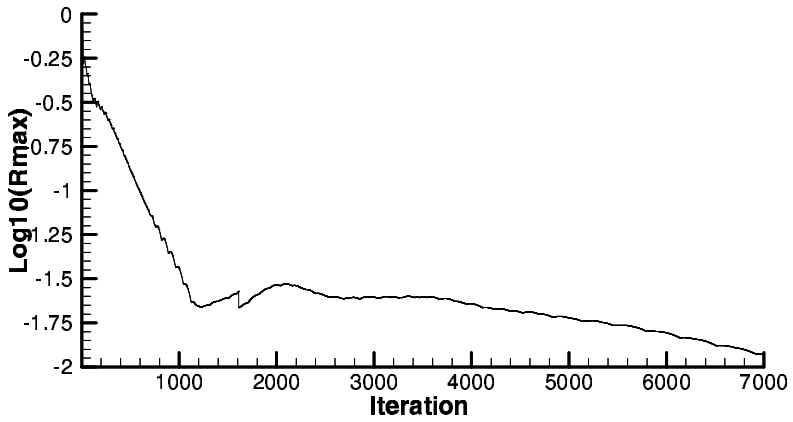


Figure 4.15: Residue plot for transonic flow past ONERA-M6 wing

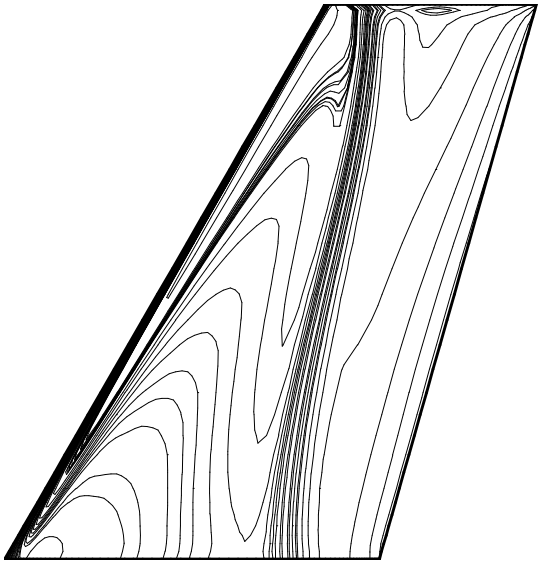


Figure 4.16: Pressure Contours on Upper surface of ONERA-M6 wing

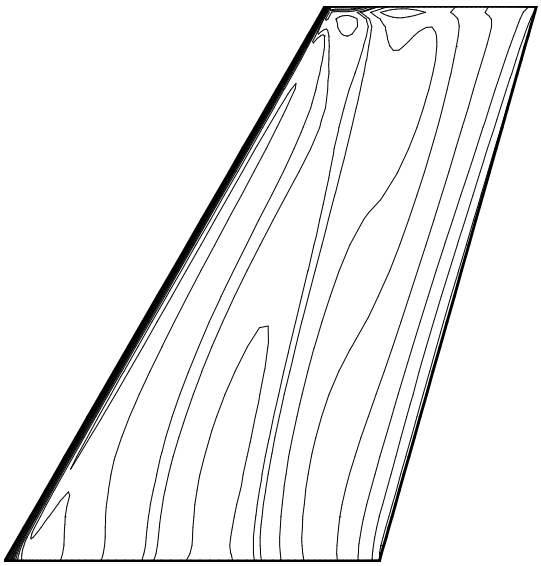


Figure 4.17: Pressure Contours on lower surface of ONERA-M6 wing

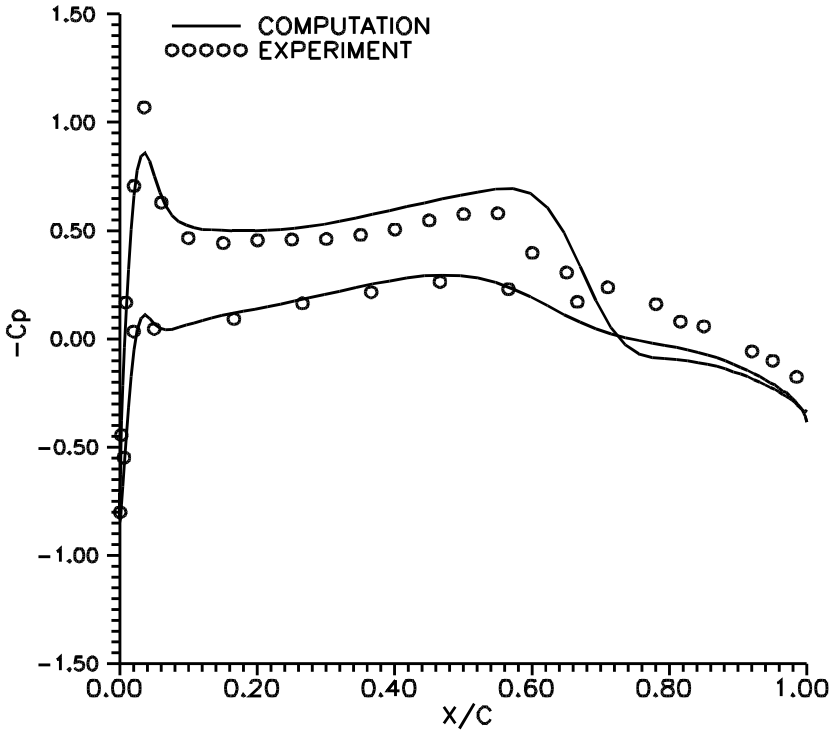


Figure 4.18: Cp distribution on ONERA-M6 wing at 20% span

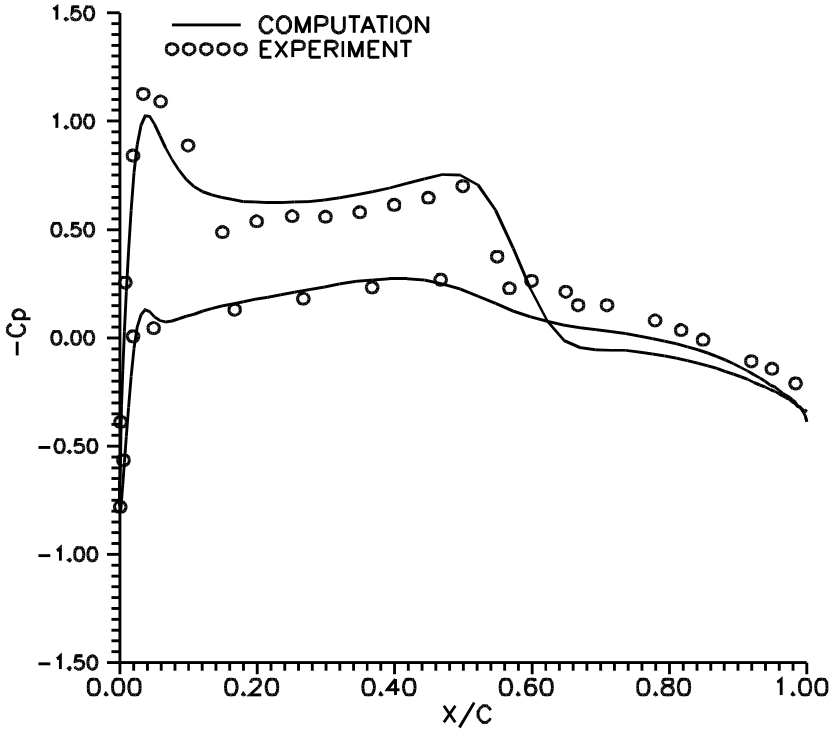


Figure 4.19: Cp distribution on ONERA-M6 wing at 44% span

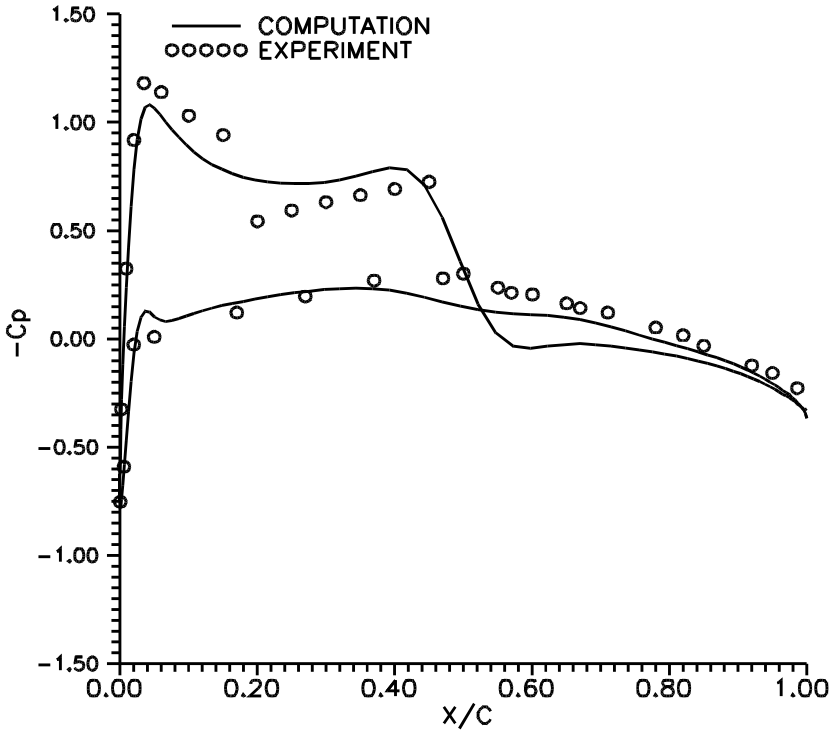


Figure 4.20: Cp distribution on ONERA-M6 wing at 65% span

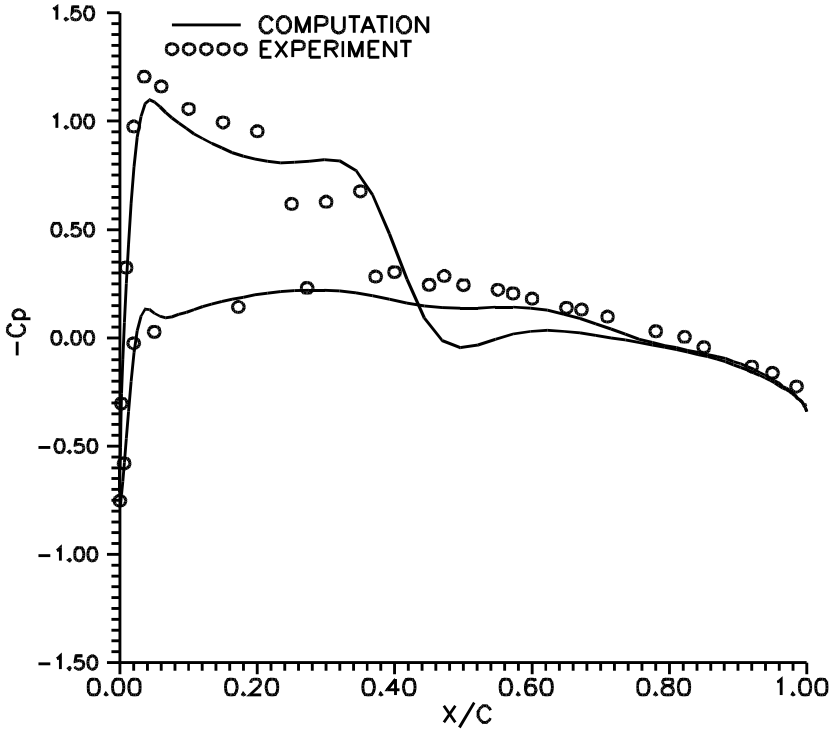


Figure 4.21: Cp distribution on ONERA-M6 wing at 80% span

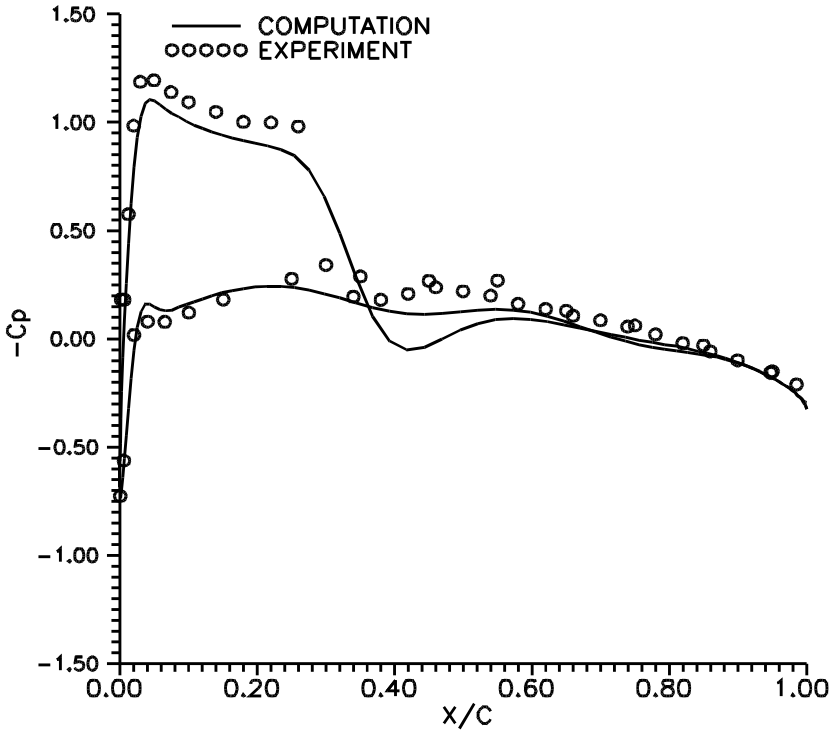


Figure 4.22: Cp distribution on ONERA-M6 wing at 90% span

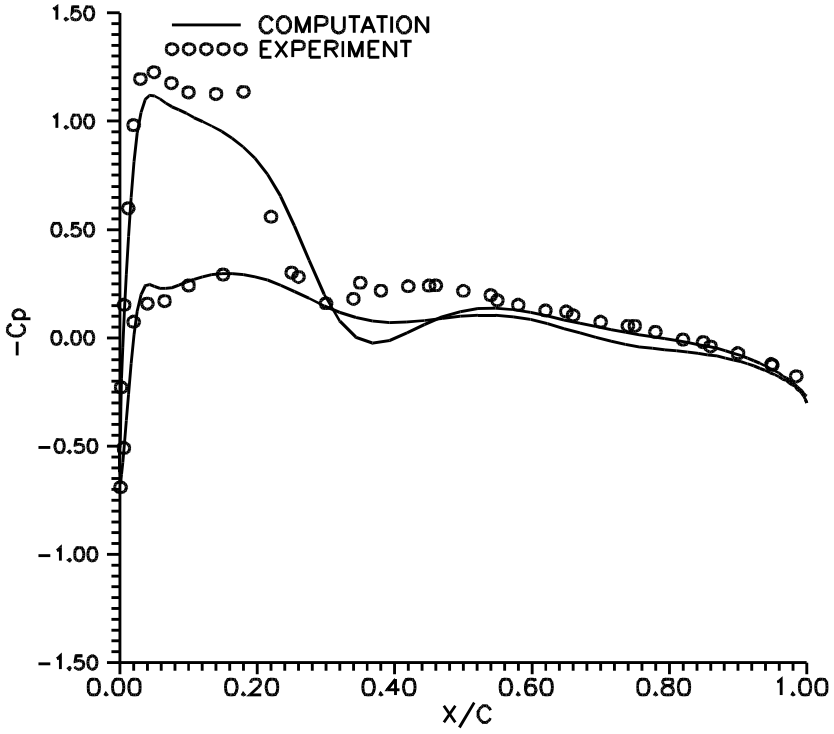


Figure 4.23: Cp distribution on ONERA-M6 wing at 95% span

4.5 Hypersonic flow past a generic flight vehicle

In this section we consider hypersonic flow past a generic flight vehicle. The configuration consists of bluntcone-cylinder-flare with lifting surfaces. For this configuration we have generated the distribution of points using two block structured grid generated around the given configuration. Fig. 4.24 shows a typical view of the point distribution. The first block of the grid is an axisymmetric 3D grid over the bluntcone-cylinder portion. The second block consists of stacked 3D grid beyond the bluntcone-cylinder portion. We have used two grids, Grid1 of size (64 $\times$ 145 $\times$ 15) and Grid2 of size (64 $\times$ 145 $\times$ 30) for all our computations. Grid1 contains approximately 1.4L(140,000) points and Grid2 has about 2.8L(240,000) points. The Mach numbers considered are 4 and 6 with angle of attack 0^o and 2^o for both these Mach numbers.

Tables 4.2 to 4.5 gives a comparison of the computed lift, drag and moment coefficients with the experiment[83]. The moment coefficients are calculated about the nose point. We observe from the tables that for $\alpha = 0^o$ for all mach numbers, the moment and lift coefficients are zero and these are quite satisfactorily predicted by the code. Also overall the comparison of the predicted values for all coefficients is quite satisfactory. We can also notice that the code slightly overpredicts all the coefficients for the higher grid size. However a through grid independence study is needed to exactly establish the deviations the code would predict from the experiment values. In the present work we have not gone into the complete exercise of validation and certification. The primary objective here is to demonstrate the satisfactory performance of 3D LSKUM code, a grid refinement study which is a part of VVC(verification, validation and certification) exercise can always be done. Figures 4.25 to 4.51 show various details of the computed results. These figures show the histories for the various computed aerodynamic coefficients, centreline plots of mach number, surface mach number contours and also mach contours in azimuthal planes. The plots of the histories of the various coefficients in general indicate, that the drag and the lift coefficients settle well ahead of the moment coefficient. Also for non zero angle of attack cases, the convergence in general is not as fast as that for zero angle of attack cases. The centreline plots also show that the predicted shock positions are always in agreement with the values given by the emperical relation due to Billing[2]. All the contour plots also show that the solutions are captured well without any undesirable oscillations.

	C_N	C_A	C_M
CFD-GRID1	0.0	0.178	0.0
CFD-GRID2	0.0	0.186	0.0
EXPT	0.0	0.153	0.0

Table 4.2: Comparison of C_N, C_A & C_M for $M = 4.0$ $\alpha = 0.0^o$

	C_N	C_A	C_M
CFD-GRID1	0.081	0.155	0.218
CFD-GRID2	0.101	0.186	0.291
EXPT	0.092	0.157	0.268

Table 4.3: Comparison of C_N, C_A & C_M for $M = 4.0$ $\alpha = 2.0^o$

	C_N	C_A	C_M
CFD-GRID1	0.0	0.160	0.0
CFD-GRID2	0.0	0.172	0.0
EXPT	0.0	0.137	0.0

Table 4.4: Comparison of C_N, C_A & C_M for $M = 6.0$ $\alpha = 0.0^o$

	C_N	C_A	C_M
CFD-GRID1	0.076	0.163	0.208
CFD-GRID2	0.080	0.176	0.257
EXPT	0.078	0.140	0.215

Table 4.5: Comparison of C_N, C_A & C_M for $M = 6.0$ $\alpha = 2.0^o$

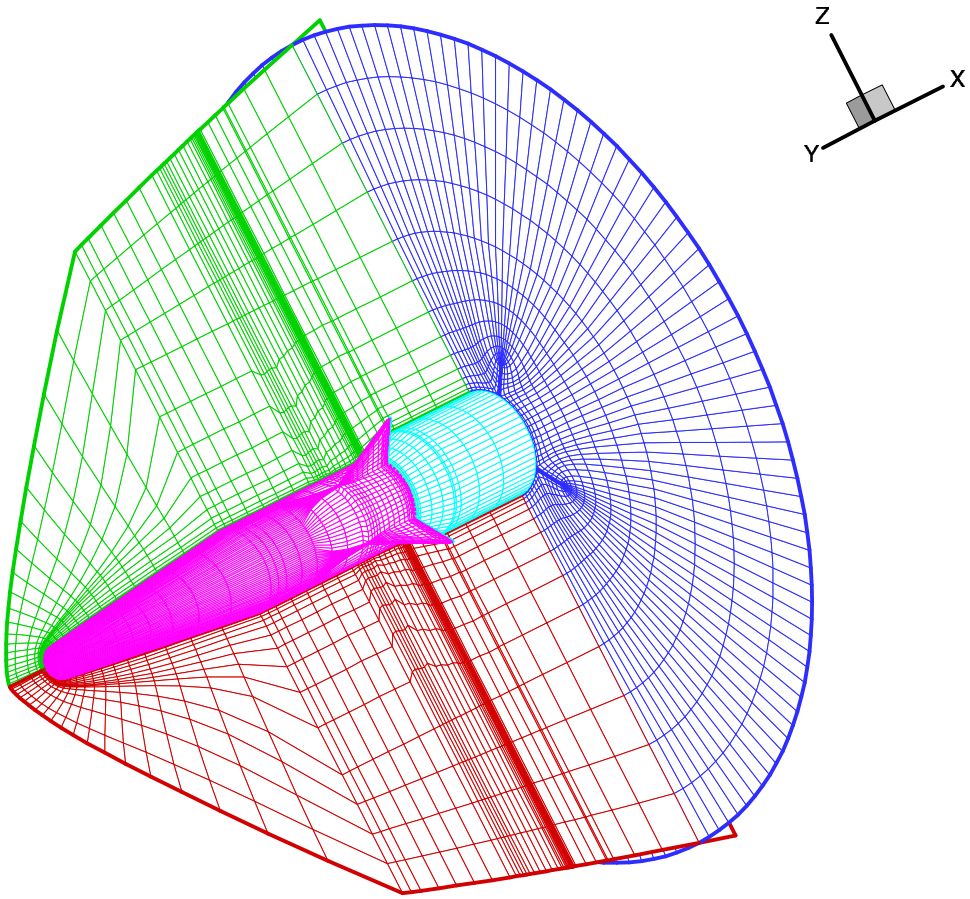


Figure 4.24: A Partial view of the grid for the typical flight vehicle

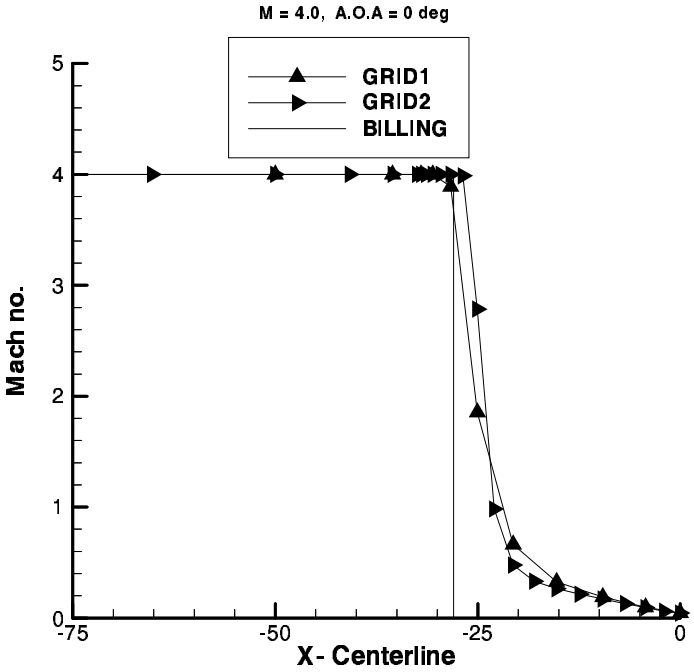


Figure 4.25: Centerline Mach no.plot: $M = 4.0 \;\; \alpha = 0^o$

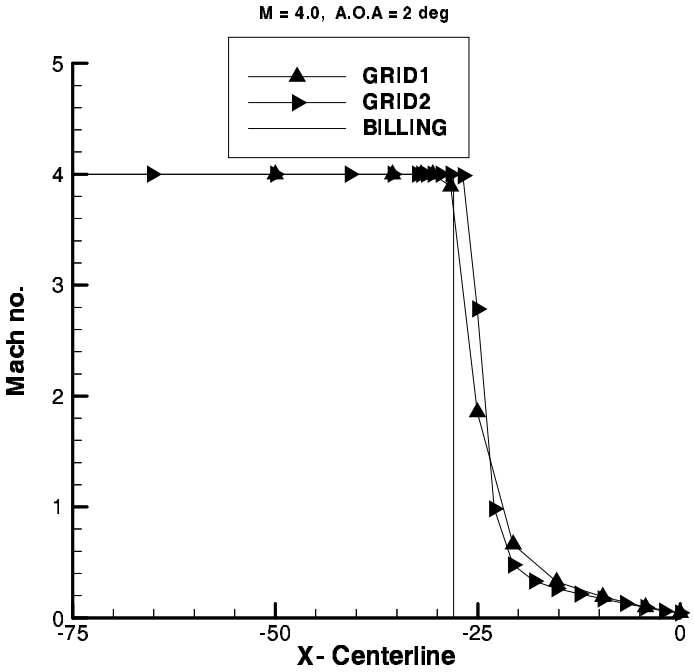


Figure 4.26: Centerline Mach no.plot: $M = 4.0 \;\; \alpha = 2^o$

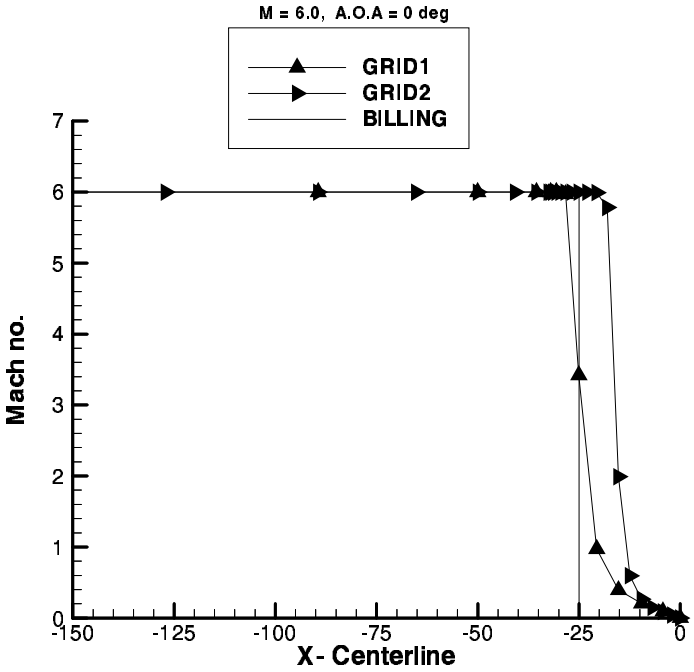


Figure 4.27: Centerline Mach no.plot: $M = 6.0$ $\alpha = 0^o$

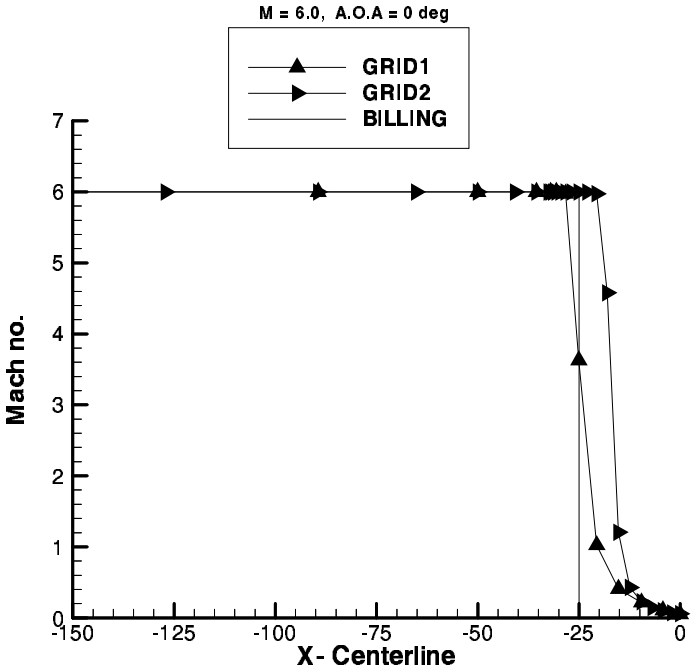


Figure 4.28: Centerline Mach no.plot: $M = 6.0$ $\alpha = 2^o$

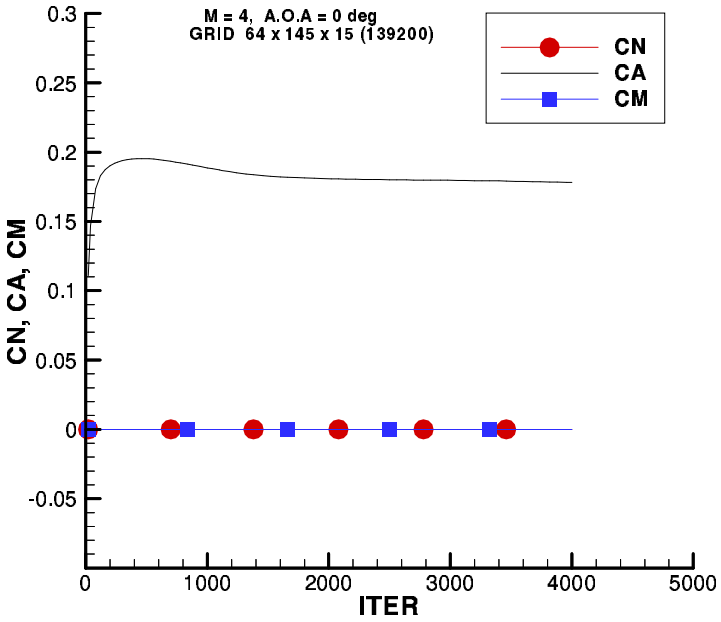


Figure 4.29: Grid1 : Coefficients plot : $M = 4.0 \;\; \alpha = 0^o$

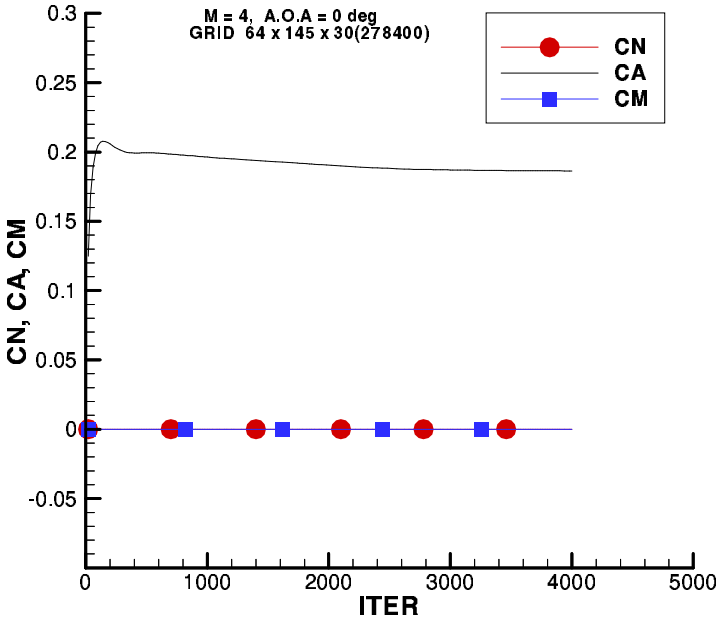


Figure 4.30: Grid2 : Coefficients plot : $M = 4.0 \;\; \alpha = 0^o$

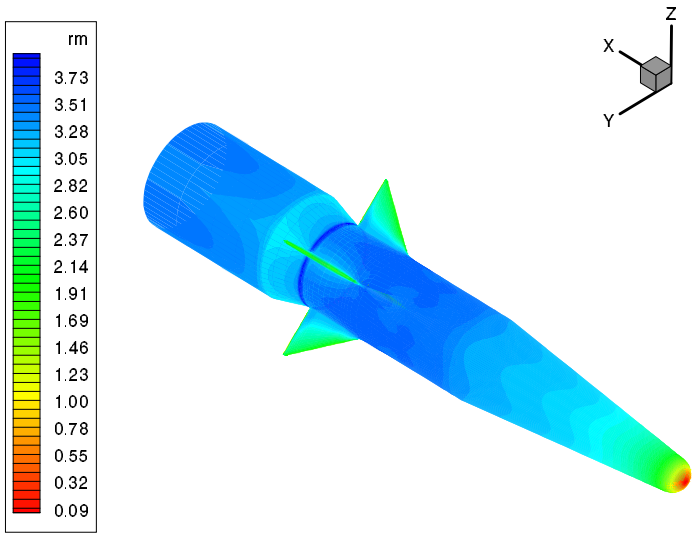


Figure 4.31: Grid1 : Surface Mach contours for $M = 4.0 \;\; \alpha = 0^o$

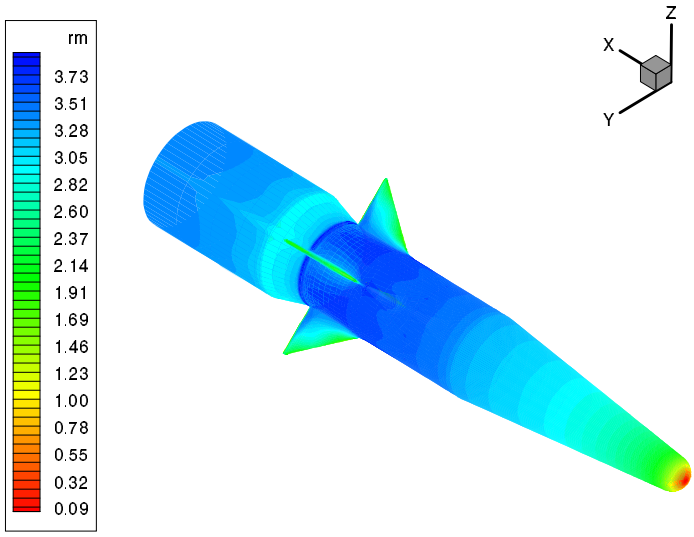


Figure 4.32: Grid2 : Surface Mach contours for $M = 4.0 \;\; \alpha = 0^o$

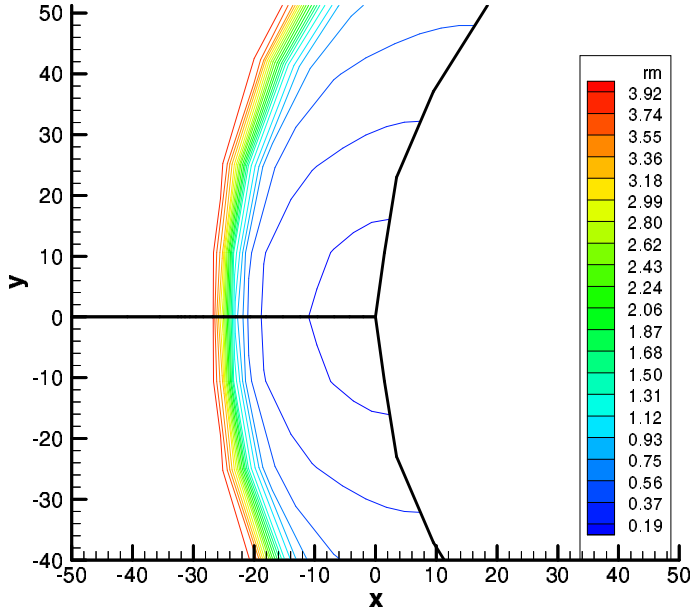


Figure 4.33: Grid2 : Mach contours near the nose : $M = 4.0 \;\; \alpha = 0^o$

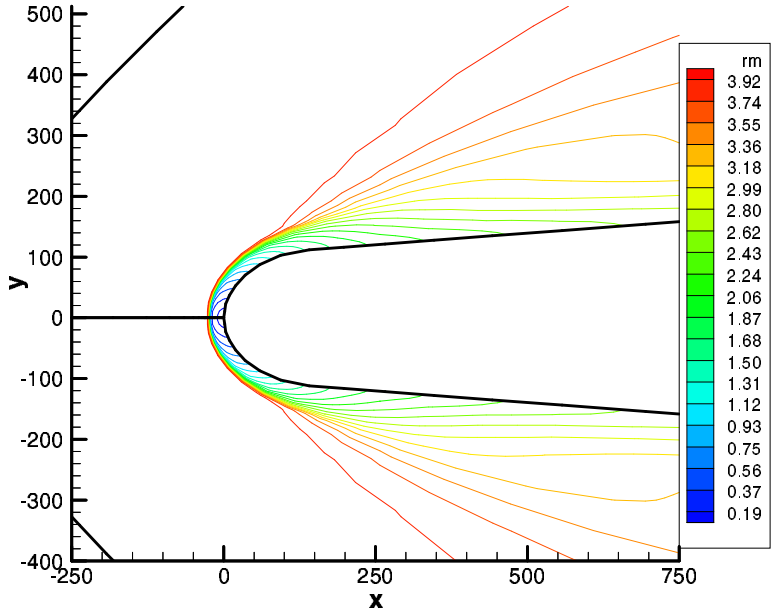


Figure 4.34: Grid2 : Mach contours in an azimuthal plane: $M = 4.0 \;\; \alpha = 0^o$

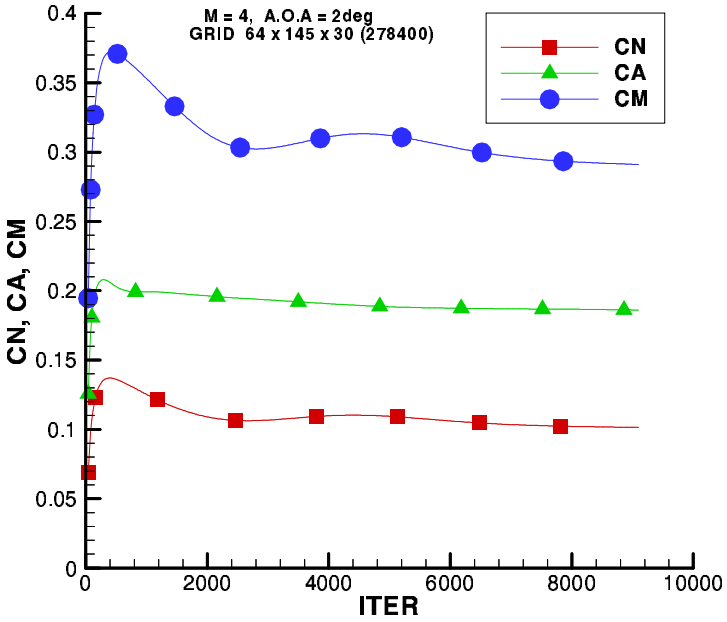


Figure 4.35: Grid2 : Coefficients plot : $M = 4.0 \;\; \alpha = 2^o$

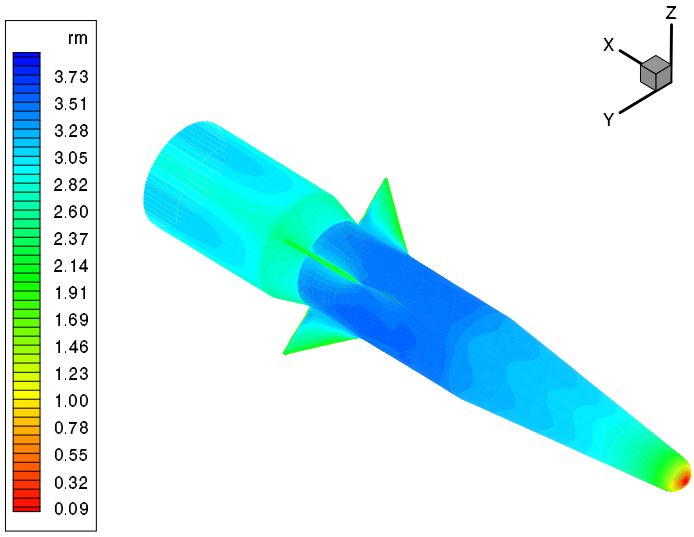


Figure 4.36: Grid1 : Surface Mach contours for $M = 4.0 \;\; \alpha = 2^o$

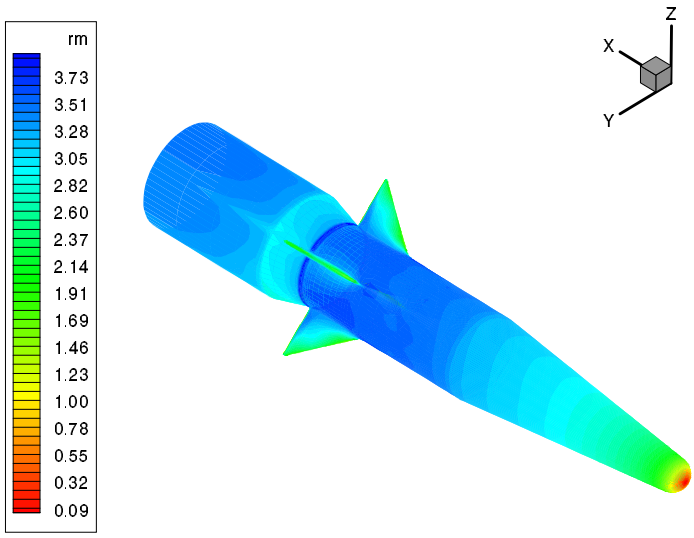


Figure 4.37: Grid2 : Surface Mach contours for $M = 4.0 \;\; \alpha = 2^o$

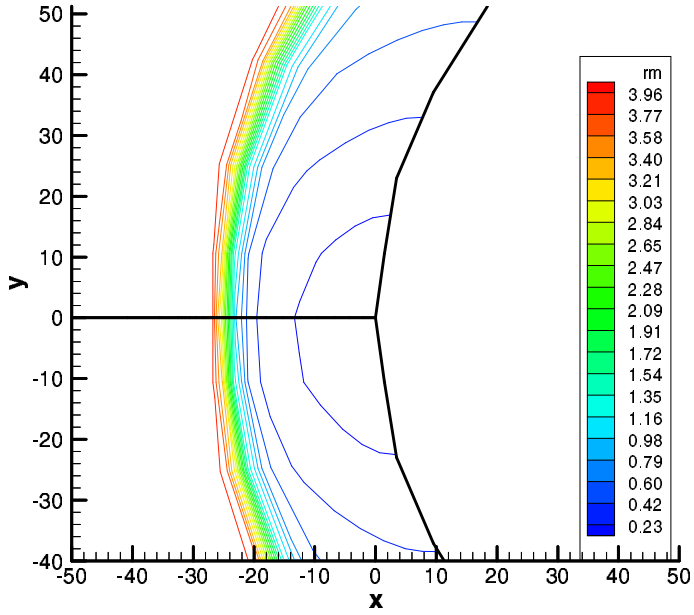


Figure 4.38: Grid2 : Mach contours near the nose : $M = 4.0 \;\; \alpha = 2^o$

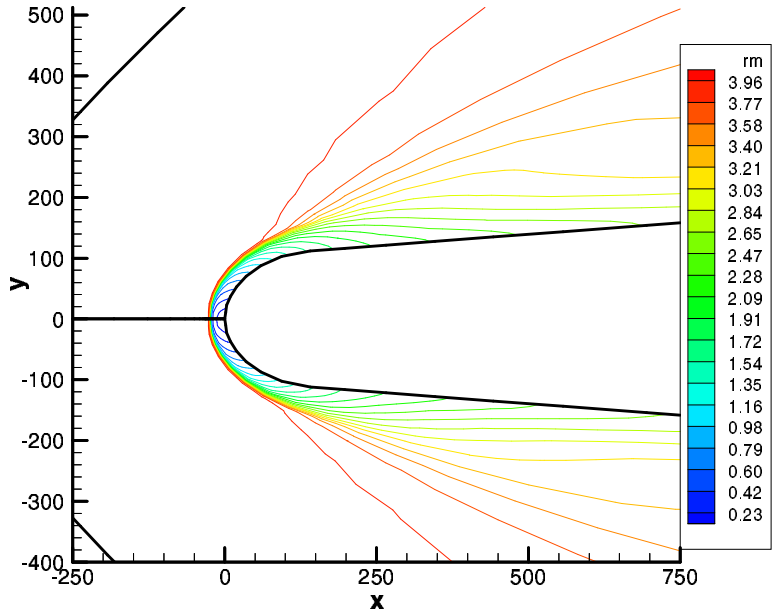


Figure 4.39: Grid2 : Mach contours in an azimuthal plane: $M = 4.0 \;\; \alpha = 2^o$

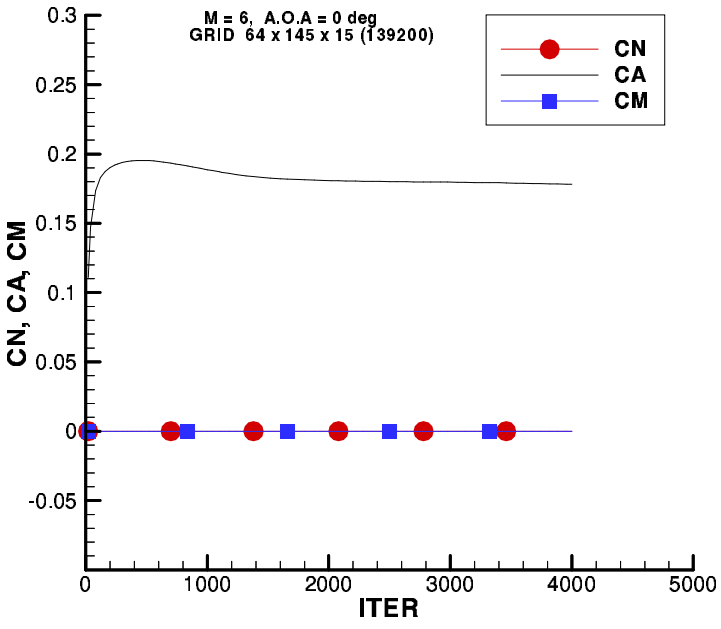


Figure 4.40: Grid1 : Coefficients plot : $M = 6.0 \;\; \alpha = 0^o$

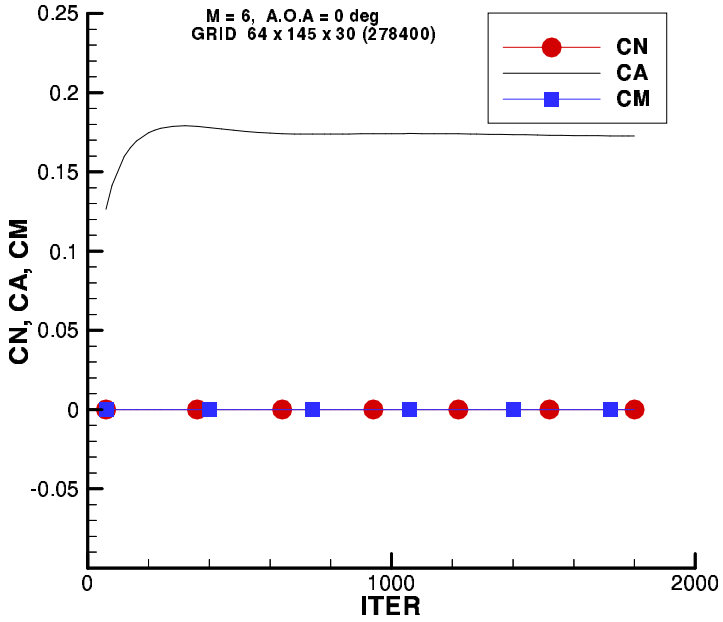


Figure 4.41: Grid2 : Coefficients plot : $M = 6.0 \;\; \alpha = 0^o$

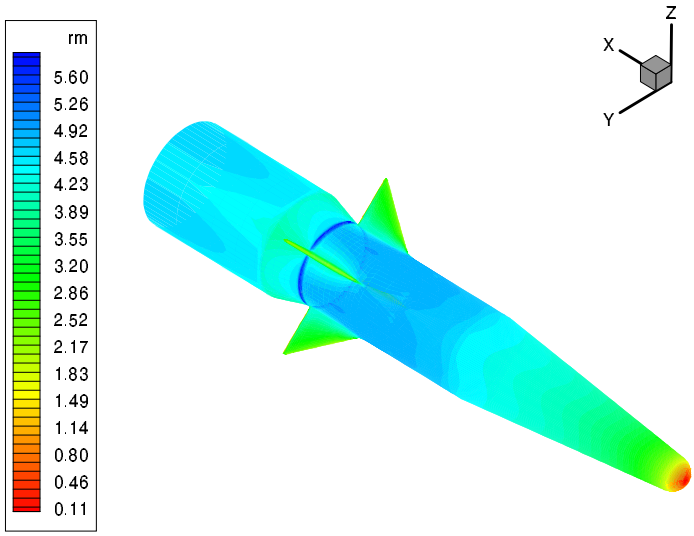


Figure 4.42: Grid1 : Surface Mach contours for $M = 6.0 \;\; \alpha = 0^o$

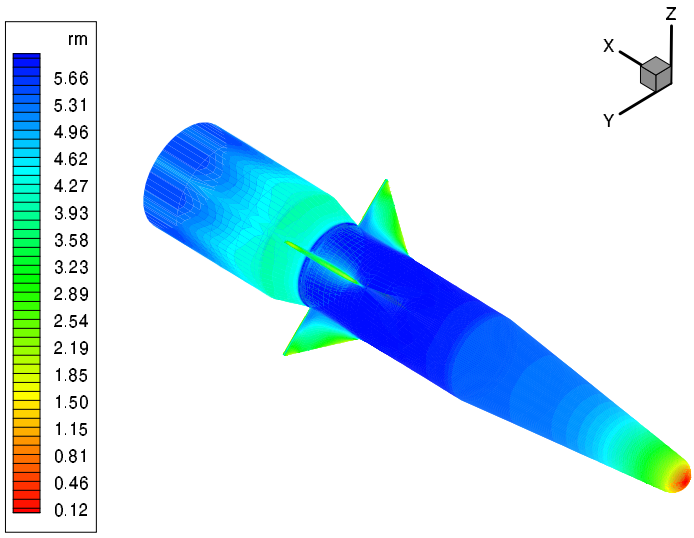


Figure 4.43: Grid2 : Surface Mach contours for $M = 6.0 \;\; \alpha = 0^o$

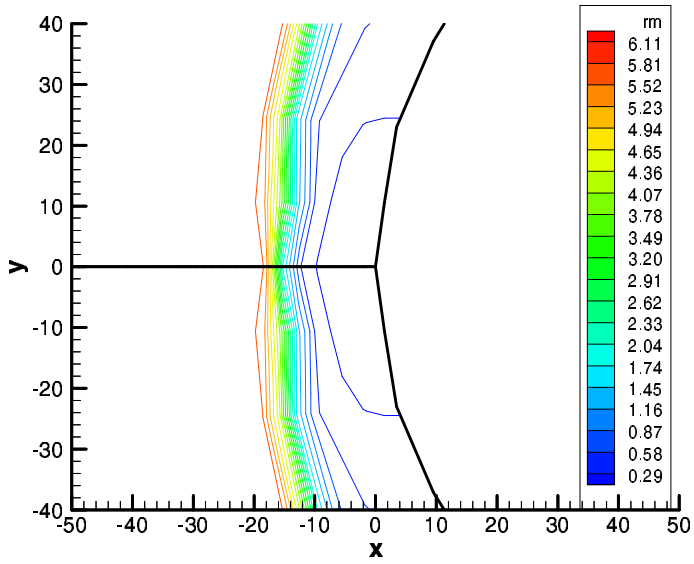


Figure 4.44: Grid2 : Mach contours near the nose : $M = 6.0 \;\; \alpha = 0^o$

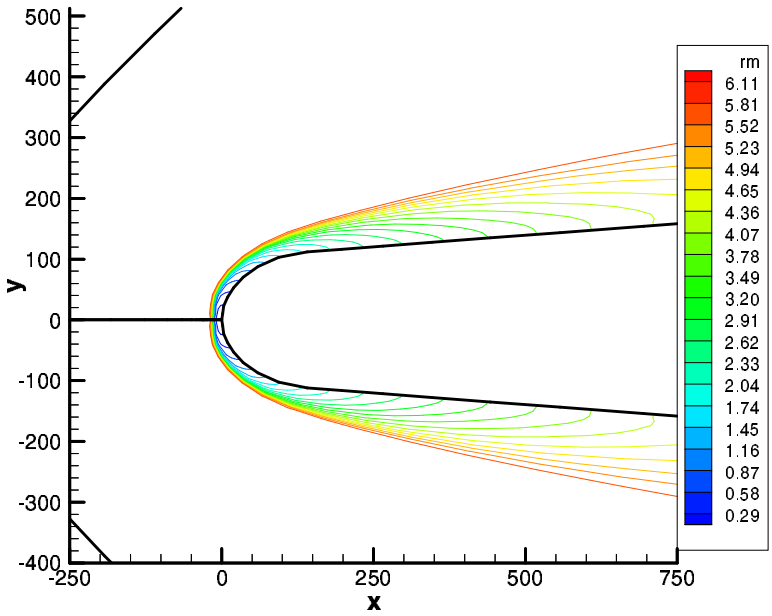


Figure 4.45: Grid2 : Mach contours in an azimuthal plane: $M = 6.0 \;\; \alpha = 0^o$

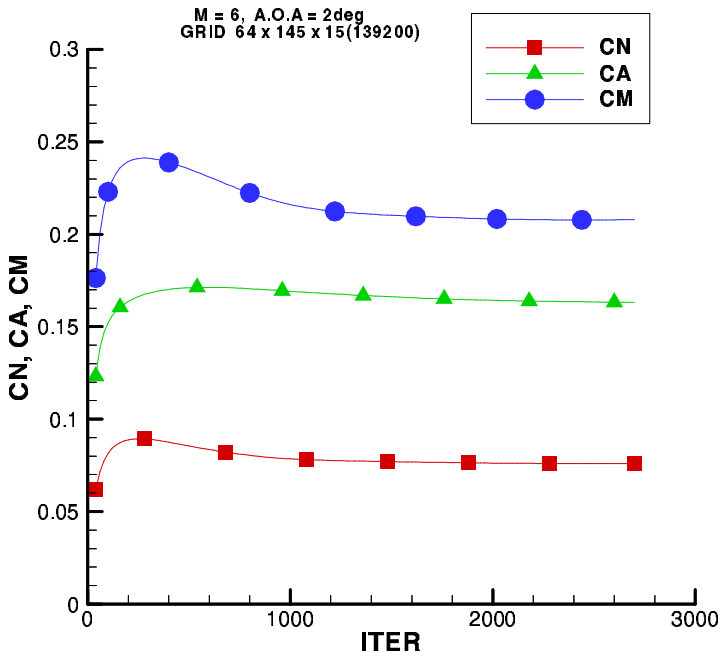


Figure 4.46: Grid1 : Coefficients plot : $M = 6.0 \;\; \alpha = 2^o$

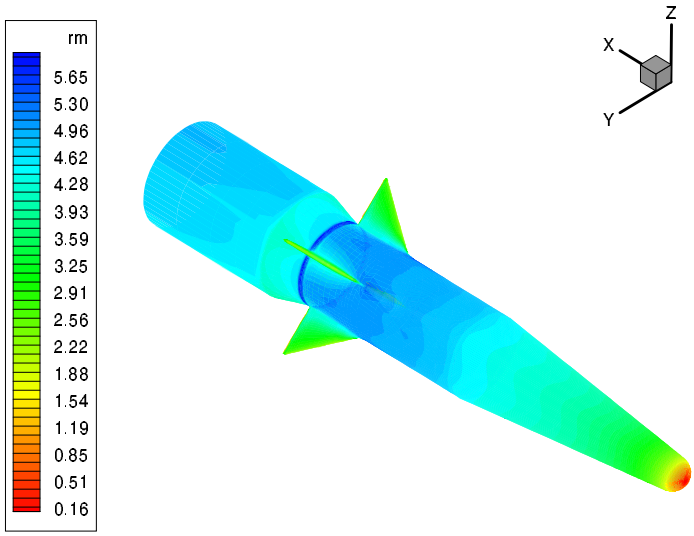


Figure 4.47: Grid1 : Surface Mach contours for $M = 6.0 \;\; \alpha = 2^o$

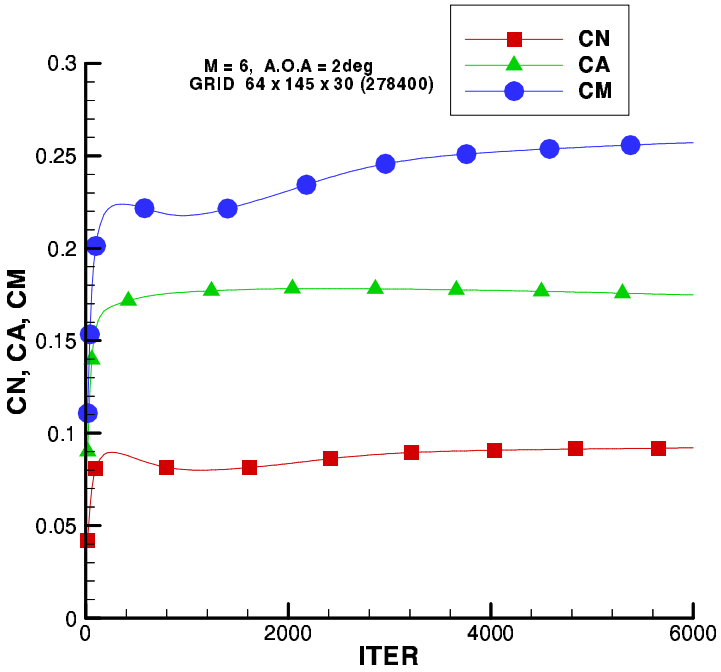


Figure 4.48: Grid2 : Coefficients plot : $M = 6.0 \;\; \alpha = 2^o$

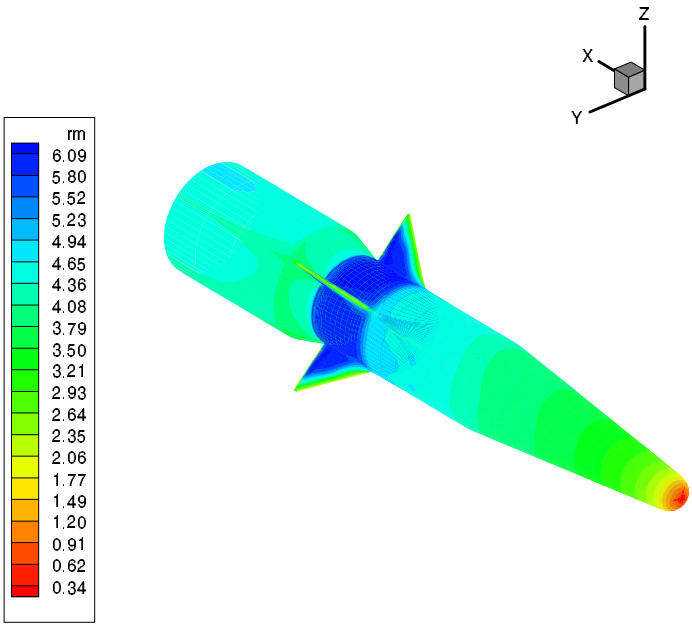


Figure 4.49: Grid2 : Surface Mach contours for $M = 6.0 \;\; \alpha = 2^o$

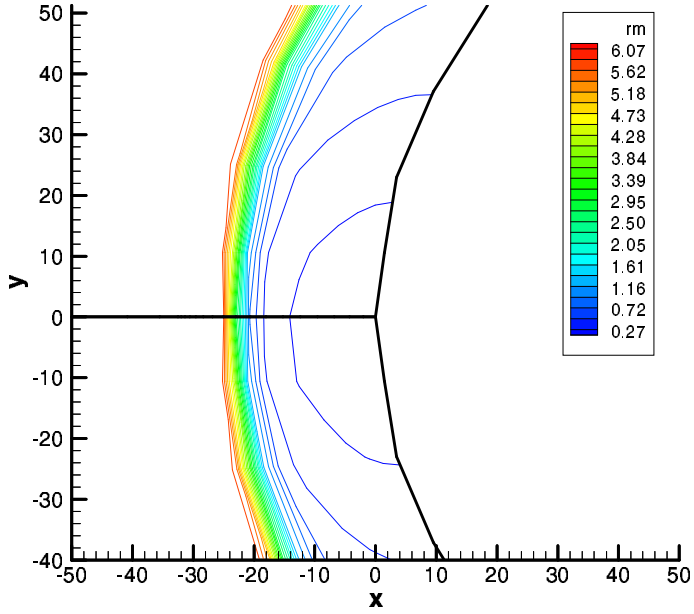


Figure 4.50: Grid2 : Mach contours in an azimuthal plane: $M = 6.0$ $\alpha = 2^o$

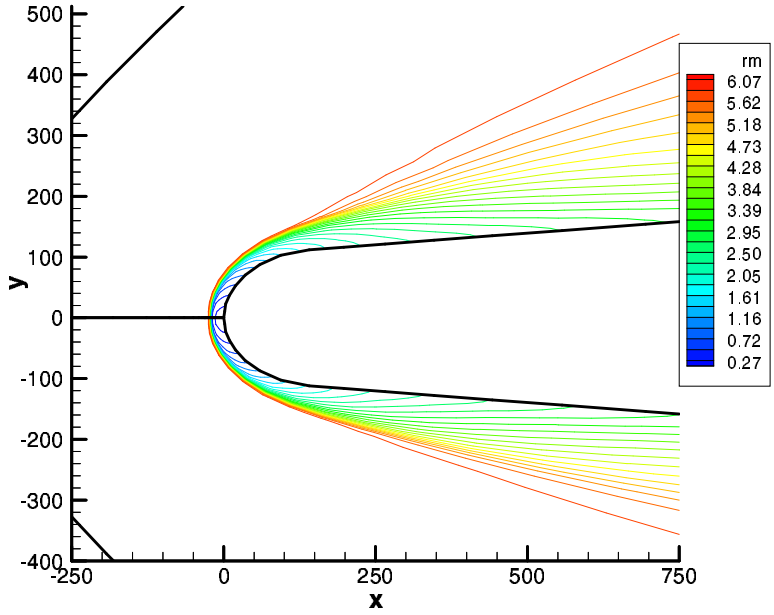


Figure 4.51: Grid2 : Mach contours in an azimuthal plane: $M = 6.0$ $\alpha = 2^o$

4.6 Conclusions

In this chapter we have presented the results obtained using 3-D LSKUM. In the first case we considered the problem of numerical simulation of spherically symmetric intense blast wave propagation in air. By succesfully computing this case we establised the capability of the solver to work even under extreme flow conditions which involved very high temperature gradients. Even though the problem consists of a spherically symmetric propagating blast wave, we have captured this feature using point distribution from a Cartesian grid. This ability of LSKUM to capture spherically symmetric blast wave on a cartesian mesh is probably because, the spatial derivatives at a node are determined using the entire data surrouding the node thus avoiding to a large extent the influence of coordinate frame chosen. A very good comparison of the predicted radius of the blast wave at different time instants with the experimental and the analytical values of G.I.Taylor was also obtained. For all these computations we have used the new Kinetic Outer Boundary Condition.

In the first test case the problem did not involve any solid boundaries. In order to establish the working of the method for problems involving solid boundary condition, we have considered the case of supersonic flow past a hemisphere. For this case we obtained good comparison with the Billings values for the predicted shock stand off distance. Also the Mach contours were captured without any undesirable oscillations.

We then considered the full 3-D problem of transonic flow past ONERA-M6[3] wing. In general 3-D transonic flows are more difficult to compute. Also this is a standard test case studied by many investigators[16, 17]. The computed results for this case successfully captured the lambda shock on the upper surface which is an important feature of this problem. For this case a good comparison of the computed pressure coefficients with the experimental values at different span locations were obtained.

Having demonstrated the satisfactory performance of the solver for a variety of test cases, we finally computed hypersonic flow past a complex generic flight vehicle. For this case we obtained very encouraging results. A good comparison of the various computed aerodynamic coefficients with the experimental values were obtained.

After demonstrating the power of LSKUM to work on any type of mesh in 2-D and 3-D flows, it could be very interesting if we could extend this method for moving grids. This could then not only operate on any type of mesh, but can work on any mesh with arbitrary grid velocities. It is this which motivated us to develop a new method LSKUM-MG which can work on moving grids. In the next chapter we present the formulation for LSKUM on moving grids for 1-D and 2-D problems. Also we present an innovative approach for treatment of boundary conditions for moving surfaces. Finally results for 1-D and 2-D moving grid problems are presented.

Chapter 5

Further developments on LSKUM

In this chapter we present two further extensions to LSKUM. First we present the formulation and results for LSKUM on moving grids. In the second extension, we give a new novel higher order approach for LSKUM. This method gives an alternative way to obtain higher order accurate approximations to derivatives in the least squares framework on relatively compact stencils. This is different from the usual two step defect correction method, where the support stencil progressively expands as we demand higher order approximations for the derivatives.

After demonstrating the power of LSKUM for computing flow field around bodies with fixed boundaries, it is tempting to ask whether LSKUM can be extended to problems involving moving boundaries and moving grid. A similar extension of KFVS for moving grids in the frame work of Finite Volume Method has been succesfully done by Krishnamurthy et al.[44, 43]. Obviously LSKUM has the potential not only to operate on any type of mesh, but can work on any mesh with arbitrary grid velocities. This motivated us to develop a new method called LSKUM-MG which can work on moving grids. Extension of LSKUM to moving grids[68, 69, 42] has been done in an elegant way at the Boltzmann level itself. When the grid velocity is zero the present formulation reduces exaclty to that on a static grid. In the following sections we present the formulation for LSKUM on moving grids for 1-D and 2-D problems. Also we present an innovative approach for treatment of boundary conditions for moving surfaces. Results for 1-D and 2-D moving grid problems are also presented. The new novel higher order approach for LSKUM has been tested for 1-D shock tube problem. The thesis ends with some preliminary results for this problem.

5.1 LSKUM on Moving Grids - (LSKUM-MG)

Computation of unsteady flows is an important problem in the field of aerodynamics. Prediction of aeroelastic behaviour in industrial applications needs an accurate prediction of the unsteady pressure loads. Helicopter rotor aerodynamics is one example where unsteady flow computation is essential. Flow through turbine blades

involving stators and rotors is another typical case where unsteady flow computation is needed. All such applications in general involve moving boundaries. Computation of flows involving moving boundaries in general involves moving grids. Some of the approaches[6] involving moving grids, transform the equations of motion to a moving frame. Also the boundary condition is treated in the moving frame itself. But these methods generally require interpolation of solution from the moving grids to background static grids.

The motivation for the present work was to extend the method to moving grids without involving any transfomation of the equations of motion to a non-inertial frame. Also general formulation is required in the sense if there is no grid movement, it must automatically reduce to that for stationary grids. Such a type of formulation could then work on any grid in which each grid point could have arbitrary grid velocity. The method can be applied to multi moving surfaces and also does not need any interpolation of the solution from moving grids to background stationary grids. We first present detailed mathematical formulation of 1-D LSKUM on moving grids with some results. Then the extension of LSKUM to moving grids for 2-D problems along with results for flow past an airfoil oscillating in pitch are presented.

5.2 1-D LSKUM-MG - formulation

Consider the 1-D Boltzmann equation

$$\frac{\partial f}{\partial t} + v \ \frac{\partial f}{\partial x} = J \tag{5.1}$$

where f is the velocity distribution function, v is the molecular velocity. J represents a collision term which vanishes in the Euler limit, when f is a Maxwellian distribution. The Maxwellian distribution, F, in one dimension is given by

$$F = \frac{\rho}{I_0}\sqrt{\frac{\beta}{\pi}} \exp\left[-\beta(v-u)^2 - I/I_0\right] \tag{5.2}$$

where $\beta = 1/(2RT)$, and I_0 is the internal energy due to non-translational degrees of freedom, $I_0 = \frac{2-\gamma}{\gamma-1}RT$ and u is the fluid velocity, R is the gas constant and T is the absolute temperature of the fluid. Therefore in the Euler limit we get

$$\frac{\partial F}{\partial t} + v \ \frac{\partial F}{\partial x} = 0 \tag{5.3}$$

Now let w represent the grid velocity of any grid point. Then we can rewrite eq. (5.3) equivalently as

$$\frac{\partial F}{\partial t} + w \ \frac{\partial F}{\partial x} + (v-w) \ \frac{\partial F}{\partial x} = 0 \tag{5.4}$$

Introduce a special derivative

$$\left(\frac{dF}{dt}\right)_{moving} = \left(\frac{dF}{dt}\right)_{mov} = \left(\frac{\partial F}{\partial t} + w \ \frac{\partial F}{\partial x}\right) \tag{5.5}$$

This special derivative is a derivative following the grid point and is very similar to the usual derivative following a fluid particle used often in fluid dynamics. Further let $\bar{v} = v - w$ be the particle velocity relative to the grid point. Then eq. (5.4) can be compactly written as

$$\left(\frac{dF}{dt}\right)_{mov} + \bar{v} \ \frac{\partial F}{\partial x} = 0 \tag{5.6}$$

This equation also has some very interesting characteristics. Consider the case when the grid velocity w becomes equal to fluid velocity u. In this case we have $\bar{v} = v - u = c$, where c is known as the peculiar velocity in kinetic theory of gases. We should also note that c is a normally distributed random variable around zero mean velocity, and can take all possible values varying from $-\infty$ to $+\infty$. The special time derivative of F along the path of a grid point now becomes the total derivative along the fluid particle path $u = \frac{dx}{dt}$. Thus eq.(5.5) can then be written as

$$\left(\frac{dF}{dt}\right)_{mov} = \frac{dF}{dt} = \left(\frac{\partial F}{\partial t} + u \ \frac{\partial F}{\partial x}\right) \tag{5.7}$$

The Boltzmann equation for the above moving grid(moving with u) now can be written as

$$\frac{dF}{dt} + c \ \frac{\partial F}{\partial x} = 0 \tag{5.8}$$

This is the basic equation around which Manoj et al.[48] have developed the Kinetic Smooth particle Hydrodynamics(KSPH) method. When the grid velocity becomes zero then the above formulation reduces to that on a static grid automatically.

Let us come back to the case when the grid velocity w is arbitrary. Splitting $\bar{v}$ into positive and negative parts and discretising the time derivative to first order in the eq. (5.6), we get the update scheme for the distribution function F as

$$F^{n+1} = F^n - \Delta t\left(\frac{\bar{v} + |\bar{v}|}{2}\frac{\partial F}{\partial x} + \frac{\bar{v} - |\bar{v}|}{2}\frac{\partial F}{\partial x}\right)^n \tag{5.9}$$

where F^{n+1} represents the updated distribution at the new position of the grid, Δt represents the time step. Following the moment method strategy, We now define a moment function vector, Ψ, as

$$\Psi = \left[1, v, I + \frac{v^2}{2}\right]^T \tag{5.10}$$

and the Ψ moment of F by,

$$\langle \Psi, F \rangle \equiv \int_0^\infty dI \int_{-\infty}^{\infty} dv \, \Psi F \tag{5.11}$$

The updating of the velocity distribution is now mapped to the updating of the state vector U at the Euler level by taking Ψ moments of the eq. (5.9). Thus we have at the Euler level the state update formula

$$U^{n+1} = U^n - \Delta t \left(\frac{\partial GX_m^+}{\partial x} + \frac{\partial GX_m^-}{\partial x} \right)^n \tag{5.12}$$

where

$$\begin{aligned} \langle \Psi, F \rangle &\equiv U \\ \left\langle \Psi, \frac{\bar{v} + |\bar{v}|}{2} F \right\rangle &\equiv GX_m^+ \\ \left\langle \Psi, \frac{\bar{v} - |\bar{v}|}{2} F \right\rangle &\equiv GX_m^- \end{aligned}$$

U is the state vector given by $U = (\rho \;\; \rho u \;\; \rho e)^T$, $GX_m^{\pm}$ represent the split fluxes for a moving grid. The split fluxes $GX_m^{\pm}$[68, 69, 41, 42] are related to the usual KFVS fluxes, and are given by

$$GX_m^{\pm} = [A] GX_s^{\pm} \tag{5.13}$$

where $GX_m^{\pm}$ represent the split fluxes for a moving grid, $GX_s^{\pm}$ represent the split fluxes which are similar to those for a static grid except that the fluid velocity components in this case are relative to the grid velocity and matrix $[A]$ transforms the fluxes to those on the moving grid. The fluxes $GX_s^{\pm}$ and the transformation matrix $[A]$ are given by

$$\begin{bmatrix} GX_s^{\pm}(1) \\ GX_s^{\pm}(2) \\ GX_s^{\pm}(3) \end{bmatrix} = \begin{bmatrix} \rho \left\{ \tilde{u} \tilde{A}^{\pm} \pm \tilde{B} \right\} \\ \rho \left\{ \left(\frac{p}{\rho} + \tilde{u}^2 \right) \tilde{A}^{\pm} \pm \tilde{u} \tilde{B} \right\} \\ \left\{ \tilde{u} \tilde{A}^{\pm} (p + \rho \tilde{e}) \pm \tilde{B} \left(\frac{p}{2} + \rho \tilde{e} \right) \right\} \end{bmatrix} \tag{5.14}$$

$$[A] = \begin{bmatrix} 1 & 0 & 0 \\ w & 1 & 0 \\ \frac{w^2}{2} & w & 1 \end{bmatrix} \tag{5.15}$$

where $\tilde{u} = u - w$, $\tilde{A}^{\pm} = \frac{1 \pm \mathrm{erf}\, S}{2}$,
$\tilde{e} = \frac{p}{\rho(\gamma - 1)} + \frac{\tilde{u}^2}{2}$, $\tilde{B} = \frac{e^{-S^2}}{2\sqrt{\pi \beta}}$
$\beta = \frac{1}{2RT}$, $S = \sqrt{\beta}\, \tilde{u}$, w =grid velocity

5.2.1 Least Squares evaluation of the spatial derivatives for moving grid

Consider an 1-D grid as shown in fig 5.2 Let o(primary node) be any node at which we want to update the solution. Further w_o is the grid velocity of node o and w_i is the grid point velocity of any point i in the neighbourhood of the point o. We can observe from eqn. (5.12) that in order to obtain a first order update to the solution at point o we need to evaluate $\langle \Psi, \bar{v}_o F_{x_o} \rangle$. Here v_o is the molecular velocity relative to grid point velocity w_o. For 1-D grid, the least squares approximation to F_{x_o} is given by,

$$F^{(1)}_{x_o} = \frac{\sum \Delta x_i \Delta F_i}{\sum {\Delta x_i}^2} \quad where \quad \Delta x_i = x_i - x_o, \quad \Delta F_i = F_i - F_o. \tag{5.16}$$

We easily observe from the above formula, evaluation of $\langle \Psi, \bar{v}_o F_{x_o} \rangle$ requires the evaluation of $\langle \Psi, \bar{v}_o F_i \rangle$, the ψ moments of F at the neighbourhood points i of the primary node o. We refer to these neighbouring points as secondary nodes and the neighbouring points of the secondary nodes are referred to as tertiary points(see fig. 5.2). Hence we need to evaluate $\langle \Psi, \bar{v}_o F_i \rangle$ at the secondary nodes , and therefore F_i must be expressed in terms of $\bar{v}_o$, that is,

$$F_i = \frac{\rho_i}{I_0} \sqrt{\frac{\beta_i}{\pi}} \exp\left[-\beta_i (\bar{v}_o - \bar{u}(w_o)_i)^2 - I/I_0 \right]$$

The relative fluid velocity at node i with respect to the grid velocity w_o of the primary node o is denoted by $\bar{u}(w_o)_i = (u_i - w_o)$ and must not be confused with $\bar{u}(w_i)_i = (u_i - w_i)$ which is the relative fluid velocity at node i with respect to the grid velocity w_i of the secondary node i. The important point being made here is that i represents a secondary node and the primary node is o at which spatial discretisation is done. Hence all computations must be done with respect to velocity relative to the primary grid point which is moving with velocity w_o

Let us also consider the two-step defect correction procedure for second order approximation to the derivative F_{x_o}. In the first step, the first order approximation is evaluated using equation (5.16). In the second step, we replace ΔF_i by $\Delta \tilde{F}_i$to get the second order approximation[30, 31] to the derivative,that is,

$$F^{(2)}_{x_o} = \frac{\sum \Delta x_i \Delta \tilde{F}_i}{\sum {\Delta x_i}^2} \quad \text{where} \quad \Delta \tilde{F}_i = \Delta F_i - \frac{1}{2}(\Delta x_i \Delta F_{x_i}), \quad \Delta F_{x_i} = F_{x_i} - F_{x_o} \tag{5.17}$$

From the above expressions we can see that in order to get a second order accurate solution for a moving grid, we need to obtain first order derivatives at the secondary points. This in turn requires moving fluxes at the tertiary points which as mentioned before will have to be evaluated with respect to molecular velocity relative to the primary node o.

5.2.2 Kinetic treatment of boundary condition for a moving solid wall

Consider a point P which lies on the piston moving with a velocity u_p as shown in fig 5.1. In order to update the velocity distribution function f at P we follow the principle of specular reflection used for static boundaries, except that now for a moving boundary, we consider the velocity of the molecules relative to the moving piston.

The velocity distribution function at P is constructed as the union of two half Maxwellians corresponding to the incident part and the reflected part, ie,

$$f_P = F_I \cup F_R \tag{5.18}$$

where F_I is the Maxwellian corresponding to the incident particles and F_R is the Maxwellian corresponding to the reflected particles. In the present case of moving piston we can easily see that all the particles with velocity $\bar{v} = (v - u_p) < 0$ relative to the piston will hit the wall. Thus we have

$$F_I = F_I(\bar{v}) \;\; for \;\; \bar{v} \;\; < 0$$

The reflected part is then constructed from the incident distribution using the specular reflection principle as

$$F_R = F_I(\acute{v}) \;\; for \;\; \bar{v} \;\; > 0 \;\; where \;\; \acute{v} \;\; = \;\; -\bar{v}, \;\; and \; \bar{v} = \;\; v - u_p.$$

Therefore the update for the velocity distribution function f and hence for U can be written as

$$\begin{aligned} f_P^{n+1} &= F_I^{n+1} \cup F_R^{n+1}, \;\; where \\ F_I^{n+1}(\bar{v}) &= F_i^n - \Delta t \bar{v} F_x \;\; for \;\; \bar{v} \;\; < 0 \\ F_R^{n+1}(\bar{v}) &= F_I^{n+1}(\acute{v}) \;\; ; \;\; \acute{v} \;\; = \;\; -\bar{v} \;\; for \;\; \bar{v} \;\; > 0 \\ U_P^{n+1} &= \left\langle \Psi, f_P^{n+1} \right\rangle \end{aligned}$$

The state update formula for the points on the moving boundary is given by

$$U^{n+1}(l) = U^n(l) - 2\Delta t \left\{ \frac{\partial G X_m^-(l)}{\partial x} \right\}, \;\; for \;\; l = 1\&3 \tag{5.19}$$

Notice that for the boundary points there is no update for $U(2)$ as $U(2) = \rho u_p$, u_p the piston velocity is given and hence the update of $U(2)$ is equivalent to update of $U(1) = \rho$.

The matrix $[A]$ for the point on the moving piston is given by

$$[A_{boundary}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{u_p^2}{2} & 0 & 1 \end{bmatrix} \tag{5.20}$$

5.2.3 Results for moving piston problem

The LSKUM on moving grid has been applied to 1-D piston problem. We have considered both compression as well as expansion cases(corresponding to piston moving in and out).

For the compression case ($u_p > 0$), given the pressure ratio $\frac{p_2}{p_1}$, the piston velocity and the density ratio across the propagating shock are given by the following relations[45]

$$u_p = \frac{a_\infty}{\gamma}\left(\frac{p_2}{p_1} - 1\right)\left[\frac{\frac{2\gamma}{\gamma+1}}{\frac{p_2}{p_1} + \frac{\gamma-1}{\gamma+1}}\right]^{\frac{1}{2}} \tag{5.21}$$

$$\frac{\rho_2}{\rho_1} = \frac{1 + \frac{\gamma+1}{\gamma-1}\frac{p_2}{p_1}}{\frac{\gamma+1}{\gamma-1} + \frac{p_2}{p_1}} \tag{5.22}$$

where subscript 2 represents higher pressure side and subscript 1 represents lower pressure side, p pressure, a speed of sound and ρ density. Similarly for the expansion case $u_P < 0$ we have

$$u_p = \frac{2a_\infty}{\gamma - 1}\left[(\frac{p_2}{p_1})^{\frac{\gamma-1}{2\gamma}} - 1\right] \tag{5.23}$$

$$\frac{\rho_2}{\rho_1} = \left[1 - \frac{\gamma - 1}{2}\frac{u_p}{a_\infty}\right]^{\frac{2}{\gamma-1}} \tag{5.24}$$

where subscript 2 represents lower pressure side and subscript 1 represents higher pressure side.

In the present computations for a given pressure ratio we calculate the piston velocity using above expressions. In the computations we specify this piston velocity as input parameter and obtain the pressure jump as a part of the solution. For the compression as well as the expansion cases, the grid velocity of the first grid point is equal to the piston velocity. The grid velocity of the last grid point is equal to the shock speed(C_s) for the compression case and for the expansion case it is equal to the speed of the sound(a_∞). The grid velocity for the interior points is then linearly interpolated between the end values. This is pictorially shown in fig. 5.3

Figures 5.4 and 5.5 show results for the compression case with the piston velocity $u_p = 469.8m/s$ which corresponds to a pressure jump of 5. Figures 5.4 and 5.5 show the density and pressure jump propogation at different time instants obtained by using first order LSKUM on a grid with 1001 points. It can be seen from these plots that the moving shock has been captured very well . For this case the exact density ratio is 2.818 and this is reproduced by computations.

Figures 5.6 and 5.7 show the plots for the shock propogation in terms of density and pressure jumps. These plots are for $u_p = 2640.78m/s$ which corresponds to a pressure jump of 100. Again we can see that LSKUM solver has succesfully captured

the shock propogation even for this high pressure ratio condition. The grid for this case contains 1001 points.

Figures 5.8 and 5.9 show a comparison of the first and second order calculations for a compression case with pressure ratio of 5, using 501 points in the grid. The second order solution captures the discontinuity more sharply. Figures 5.10 and 5.11 show the results for the expansion case with pressure ratio of 0.02. The grid used in this case has 1001 points. It can be observed that the smooth variation of density has been captured very well.

Table 5.1 and Table 5.2 give a quantitative comparison of the computations with the exact values for both compression and expansion cases. It can be observed clearly from these tables that there is excellent agreement with the analytical values.

Table 5.1: Compression case : Comparison with analytical values

Pressure Ratio	$u_P(m/s)$	$\left(\frac{\rho_2}{\rho_1}\right)_{exact}$	$\left(\frac{\rho_2}{\rho_1}\right)_{comp.}$
5	469.8	2.818	2.818
10	2640.8	5.669	5.669

Table 5.2: Expansion case : Comparison with analytical values

Pressure Ratio	$u_P(m/s)$	$\left(\frac{\rho_2}{\rho_1}\right)_{exact}$	$\left(\frac{\rho_2}{\rho_1}\right)_{comp.}$
0.2	-355.4	0.316	0.316
0.02	-740.74	0.061	0.061

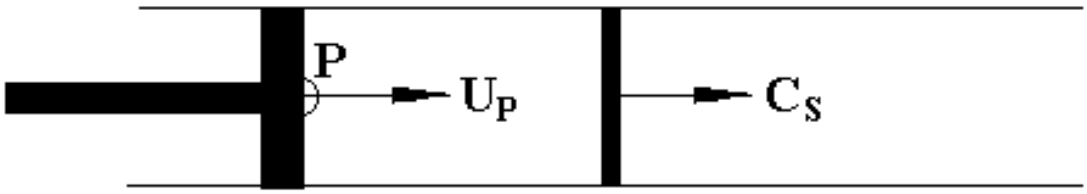


Figure 5.1: 1-D Piston problem : Schematic representation

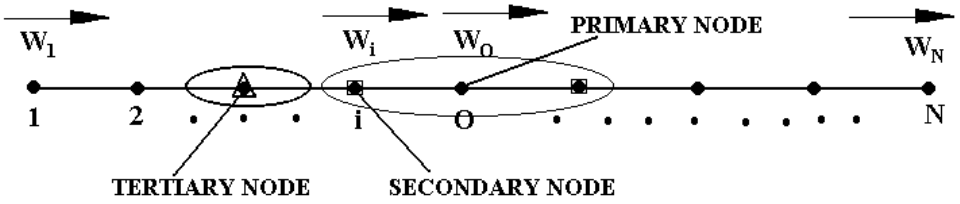


Figure 5.2: Connectivity : 1-D moving grid

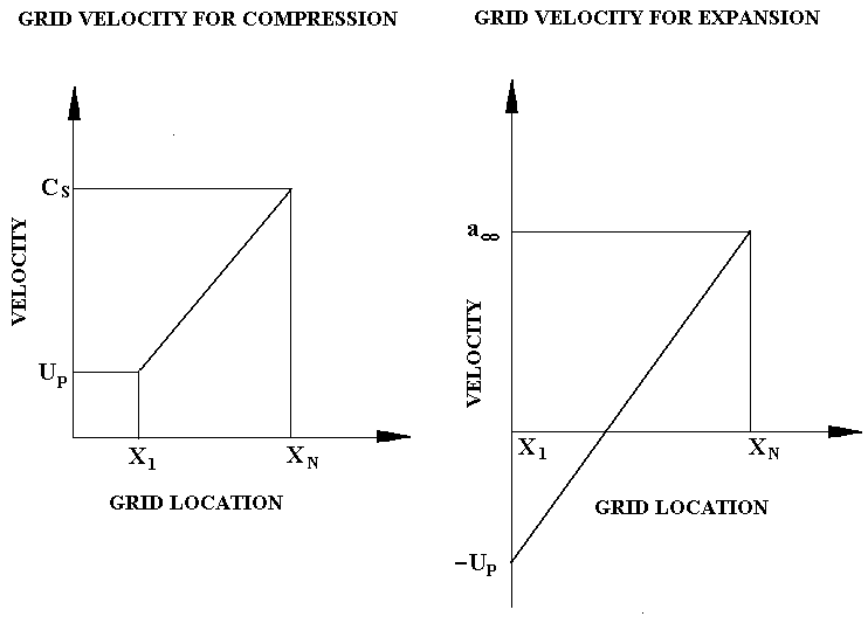


Figure 5.3: Grid velocity interpolation for 1-D piston problem

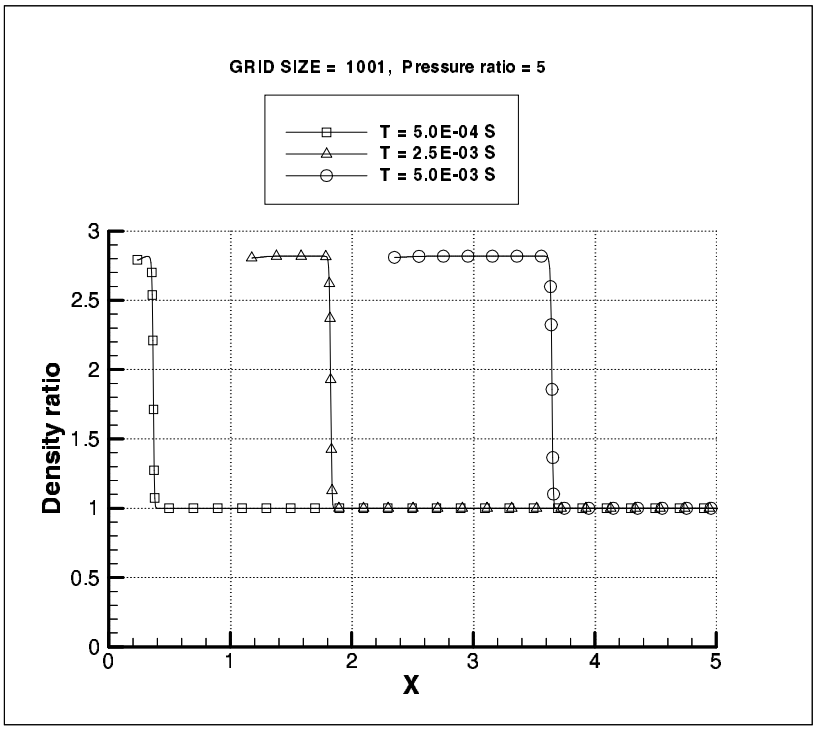


Figure 5.4: Density plot : Compression : Pressure ratio = 5

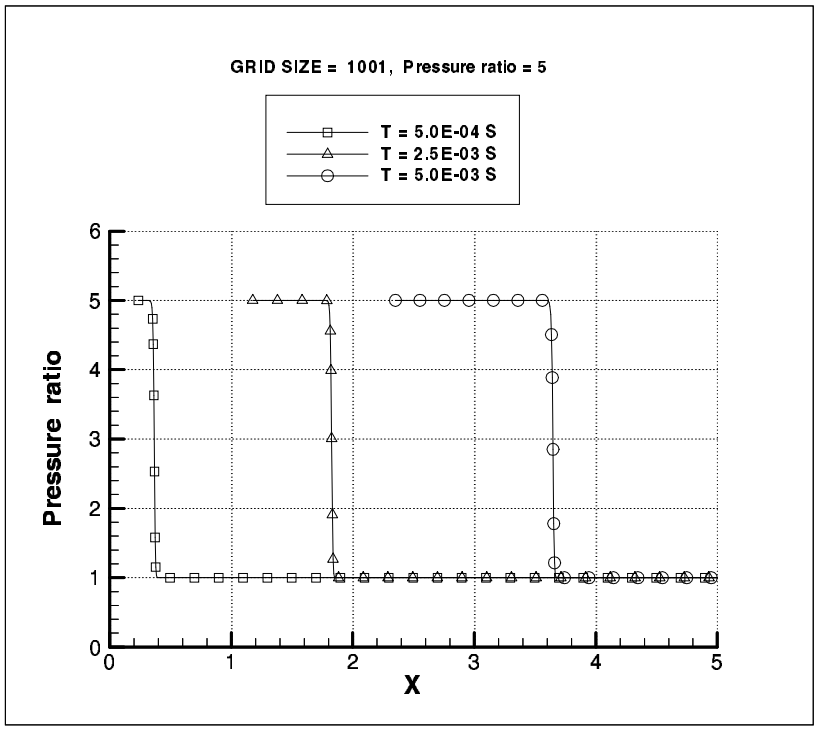


Figure 5.5: Pressure plot : Compression : Pressure ratio = 5

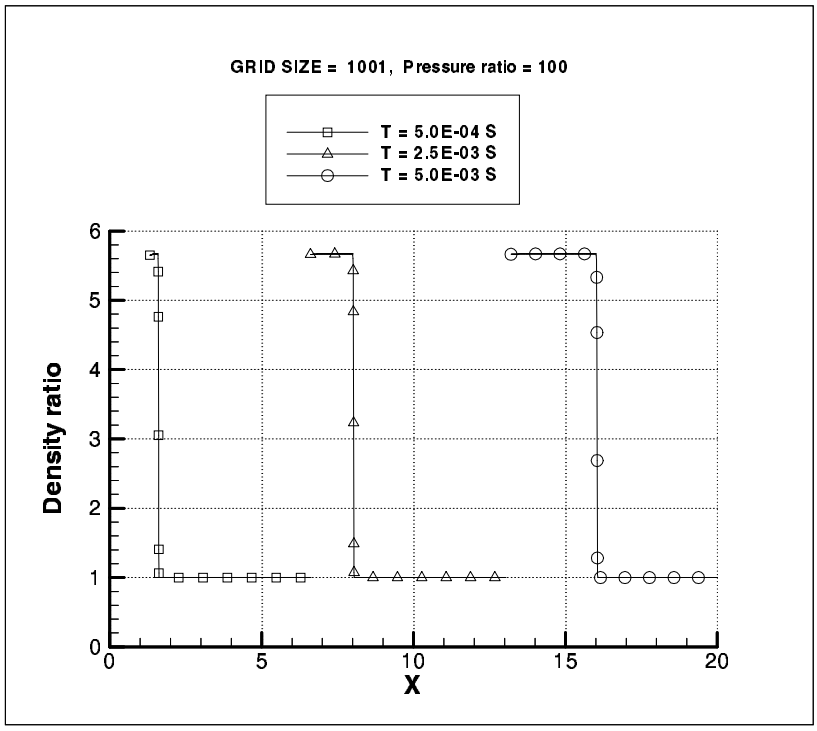


Figure 5.6: Density plot : Compression : Pressure ratio = 100

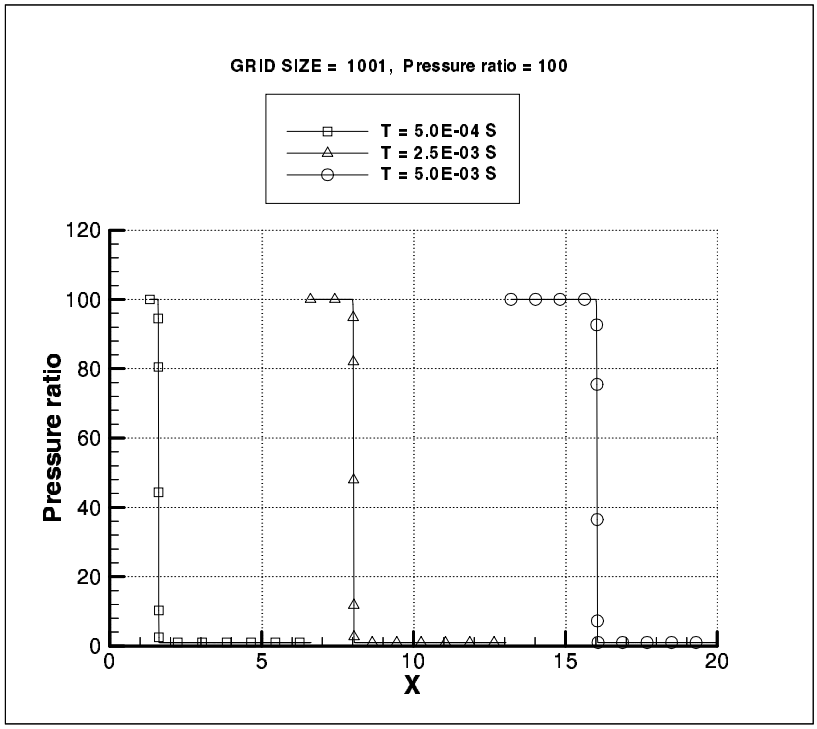


Figure 5.7: Pressure plot : Compression : Pressure ratio = 100

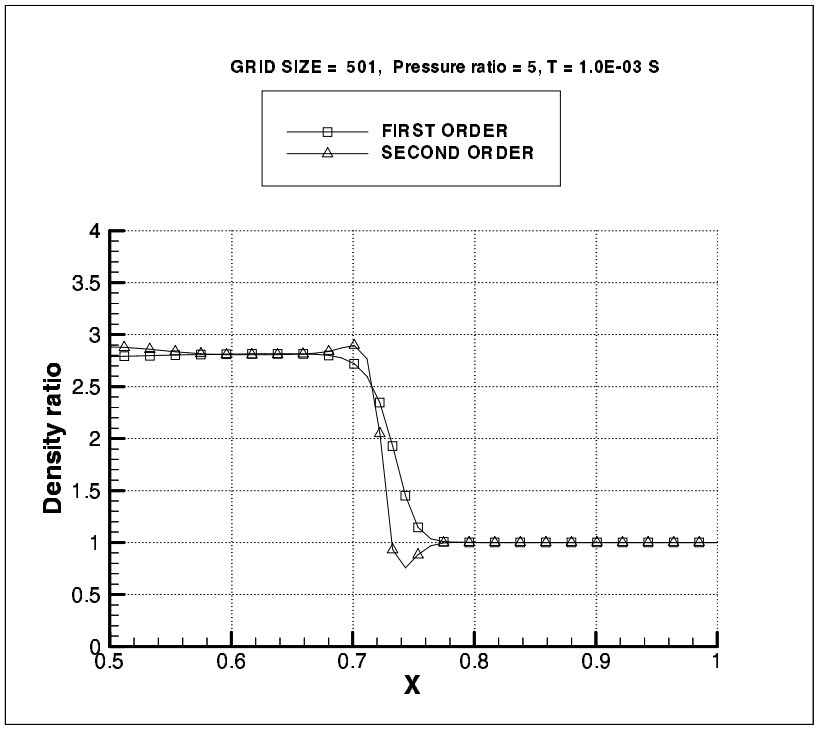


Figure 5.8: Density plot : Compression : Comparison of Ist and IInd order computations

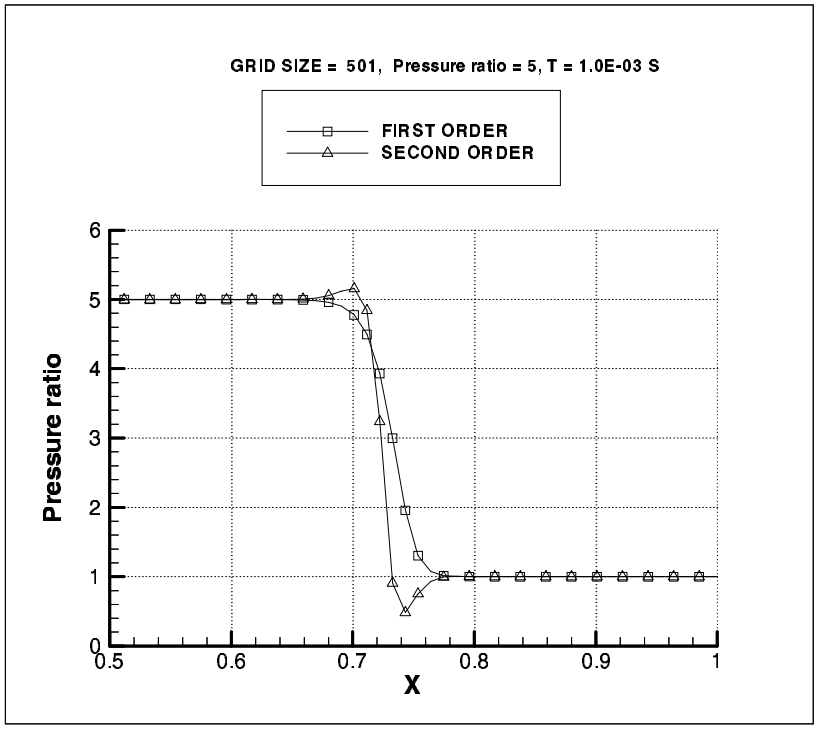


Figure 5.9: Pressure plot : Compression : Comparison of Ist and IInd order computations

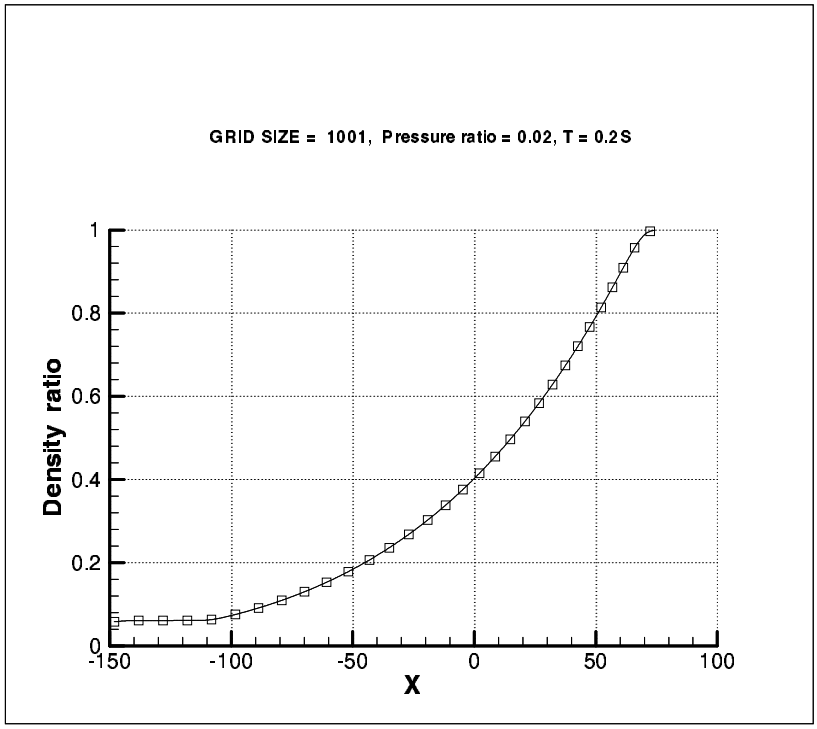


Figure 5.10: Density plot : Expansion : Pressure ratio =0.02

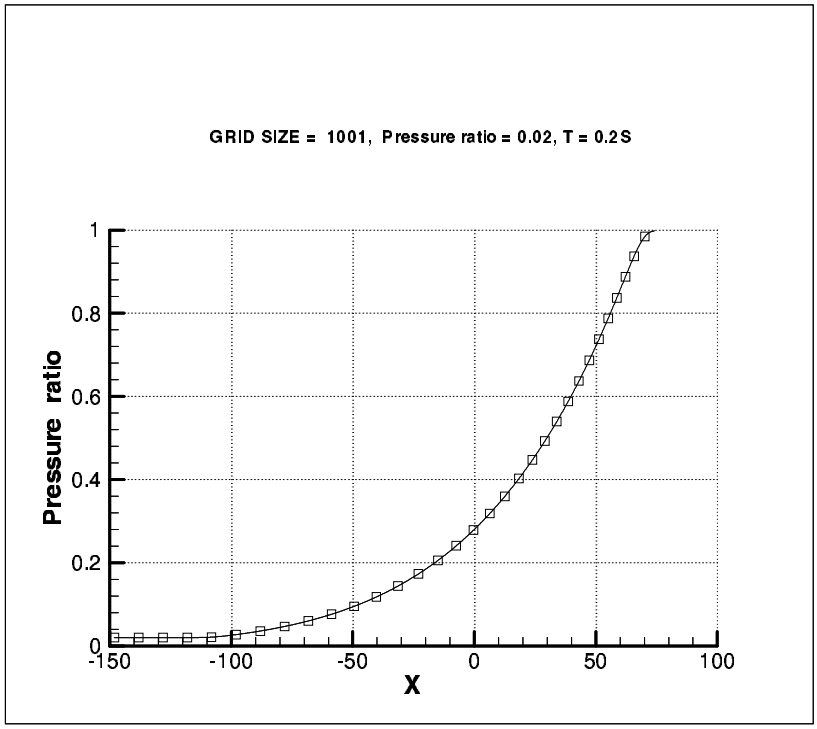


Figure 5.11: Pressure plot : Expansion : Pressure ratio =0.02

5.3 2-D : LSKUM-MG - formulation

Consider the 2-D Boltzmann equation

$$\frac{\partial f}{\partial t} + v_1 \ \frac{\partial f}{\partial x} + v_2 \ \frac{\partial f}{\partial y} = J \tag{5.25}$$

where f is the velocity distribution function, v_1 and v_2 are the Cartesian components of the molecular velocity. J represents a collision term which vanishes in the Euler limit, when f is a Maxwellian distribution. The Maxwellian distribution, F, in two dimensions is given by

$$F = \frac{\rho}{I_0}\frac{\beta}{\pi} \exp\left[-\beta(v_1 - u_1)^2 - \beta(v_2 - u_2)^2 - I/I_0\right] \tag{5.26}$$

where $\beta = 1/(2RT)$, and I_0 is the internal energy due to non-translational degrees of freedom, $I_0 = \frac{2-\gamma}{\gamma-1}RT$ and u_1 and u_2 are the Cartesian components of the fluid velocity, R is the gas constant and T is the absolute temperature of the fluid. Therefore in the Euler limit we get

$$\frac{\partial F}{\partial t} + v_1 \ \frac{\partial F}{\partial x} + v_2 \ \frac{\partial F}{\partial y} = 0 \tag{5.27}$$

Now let w_1 and w_2 represent the Cartesian components of the grid velocity of any grid point. Then we can rewrite equation (5.27) equivalently as

$$\left(\frac{\partial F}{\partial t} + w_1 \ \frac{\partial F}{\partial x} + w_2 \ \frac{\partial F}{\partial y}\right) + (v_1 - w_1) \ \frac{\partial F}{\partial x} + (v_2 - w_2) \ \frac{\partial F}{\partial y} = 0 \tag{5.28}$$

Introduce a special derivative

$$\left(\frac{dF}{dt}\right)_{moving} = \left(\frac{dF}{dt}\right)_{mov} = \left(\frac{\partial F}{\partial t} + w_1 \ \frac{\partial F}{\partial x} + w_2 \ \frac{\partial F}{\partial y}\right)$$

This is a special derivative following a grid point and further let $\bar{v_1} = v_1 - w_1$, $\bar{v_2} = v_2 - w_2$ be the components of the particle velocity relative to the grid point. Then equation (5.28) can be written as

$$\left(\frac{dF}{dt}\right)_{mov} + \bar{v_1} \ \frac{\partial F}{\partial x} + \bar{v_2} \ \frac{\partial F}{\partial y} = 0 \tag{5.29}$$

This is the 2-D Boltzmann equation for a moving grid. Consider the case when the grid velocity components w_1 and w_2 becomes equal to fluid velocity components u_1 and u_2. In this case we have $\bar{v_1} = v_1 - u_1 = c_1$, and $\bar{v_2} = v_2 - u_2 = c_2$ where c_1 and c_2 are known as the peculiar velocities in kinetic theory of gases. We should also note that c_1 and c_2 are normally distributed random variables around zero mean

velocity, and can take all possible values varying from $-\infty$ to$+\ \infty$. The special time derivative of F along the path of a grid point now becomes the total derivative along the fluid particle path $u_1 = \frac{dx}{dt}$ and $u_2 = \frac{dy}{dt}$. Thus eq.(5.29) can then be written as

$$\left(\frac{dF}{dt}\right)_{mov} = \frac{dF}{dt} = \left(\frac{\partial F}{\partial t} + u_1 \ \frac{\partial}{\partial x} + u_2 \ \frac{\partial}{\partial y}\right) \tag{5.30}$$

The Boltzmann equation for the above moving grid now can be written as

$$\frac{dF}{dt} + c_1 \ \frac{\partial F}{\partial x} + c_2 \ \frac{\partial F}{\partial y} = 0 \tag{5.31}$$

The development of 2-D KSPH[48] method starts with the above equation and uses the least squares discretisation of the spatial derivatives in eq. (5.31). The streamline upwind version of KSPH is based on locally rotated frame (s, n) where s is along a local streamline and n is the normal co-ordinate. Then eq. (5.31) in this rotated frame becomes

$$\frac{dF}{dt} + \acute{c}_1 \ \frac{\partial F}{\partial s} + \acute{c}_2 \ \frac{\partial F}{\partial n} = 0 \tag{5.32}$$

where $\acute{c}_1$ and $\acute{c}_2$ represent the components of the peculiar velocity along the local streamline direction and its normal. Upwind stencil can be used to discretise $\frac{\partial F}{\partial s}$ while full stencil can be used to discretise $\frac{\partial F}{\partial n}$. Such a discretisation is perfectly reasonable as fluid is advected along s-direction.

Let us now come back to the case when grid velocity is arbitrary. Splitting $\bar{v_1}$ and $\bar{v_2}$ into positive and negative parts and discretising the time derivative to first order in equation (5.29), we get the update scheme for the distribution F as

$$F^{n+1} = F^n - \Delta t\left(\frac{\bar{v_1} + |\bar{v_1}|}{2}\frac{\partial F}{\partial x} + \frac{\bar{v_1} - |\bar{v_1}|}{2}\frac{\partial F}{\partial x} + \frac{\bar{v_2} + |\bar{v_2}|}{2}\frac{\partial F}{\partial y} + \frac{\bar{v_2} - |\bar{v_2}|}{2}\frac{\partial F}{\partial y}\right)^n \tag{5.33}$$

where F^{n+1} represents the updated value of F at the new position of the grid, Δt represents the time step. The updating of F is now mapped to the updating of the state vector U at the Euler level by taking Ψ moments of the equation (5.33). We then obtain

$$U^{n+1} = U^n - \Delta t\left(\frac{\partial GX_m^+}{\partial x} + \frac{\partial GX_m^-}{\partial x} + \frac{\partial GY_m^+}{\partial y} + \frac{\partial GY_m^-}{\partial y}\right)^n \tag{5.34}$$

where

$$\Psi = \left[1, v_1, v_2, I + \frac{v_1^2 + v_2^2}{2}\right]^T$$

$$\langle \Psi, F \rangle \ \equiv \ \int_0^\infty dI \int_{-\infty}^\infty dv_1 \int_{-\infty}^\infty dv_2 \, \Psi F \equiv U$$

$$\begin{aligned}
\left\langle \Psi, \frac{\bar{v}_1 + |\bar{v}_1|}{2} F \right\rangle &\equiv GX_m^+ \\
\left\langle \Psi, \frac{\bar{v}_1 - |\bar{v}_1|}{2} F \right\rangle &\equiv GX_m^- \\
\left\langle \Psi, \frac{\bar{v}_2 + |\bar{v}_2|}{2} F \right\rangle &\equiv GY_m^+ \\
\left\langle \Psi, \frac{\bar{v}_2 - |\bar{v}_2|}{2} F \right\rangle &\equiv GY_m^-
\end{aligned}$$

U is the state vector given by $U = (\rho \;\; \rho u_1 \;\; \rho u_2 \;\; \rho e)^T$, $GX_m^{\pm}$ and $GY_m^{\pm}$ represents the split fluxes for a moving grid[41, 42, 68, 69]. The moving split fluxes are given by the following expressions,

$$G_m^{\pm} = [A] G_s^{\pm} \tag{5.35}$$

where $G_m^{\pm}$ represent the split fluxes for a moving grid, $G_s^{\pm}$ represent the split fluxes which are similar to those for a static grid except that the velocity components in this case are relative to the grid velocity and matrix $[A]$ transforms the fluxes to moving grid. The transformation matrix $[A]$ is given by

$$[A] \;\; = \;\; \begin{bmatrix} 1 & 0 & 0 & 0 \\ w_1 & 1 & 0 & 0 \\ w_2 & 0 & 1 & 0 \\ \frac{w_1^2 + w_2^2}{2} & w_1 & w_2 & 1 \end{bmatrix} \tag{5.36}$$

The expressions for the split fluxes $G_s^{\pm}$ are as follows.
The x-component of $G_s^{\pm}$ is given by

$$\begin{bmatrix} GX^{\pm}(1) \\ GX^{\pm}(2) \\ GX^{\pm}(3) \\ GX^{\pm}(4) \end{bmatrix} = \begin{bmatrix} \rho \left\{ \tilde{u}_1 A_1^{\pm} \pm B_1 \right\} \\ \rho \left\{ \left(\frac{p}{\rho} + \tilde{u}_1^2 \right) A_1^{\pm} \pm \tilde{u}_1 B_1 \right\} \\ \rho \left\{ \tilde{u}_1 \tilde{u}_2 A_1^{\pm} \pm \tilde{u}_2 B_1 \right\} \\ \rho \left\{ \tilde{u}_1 \left(\frac{2\gamma}{\gamma - 1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{A_1^{\pm}}{2} \pm \left(\frac{\gamma + 1}{\gamma - 1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{B_1}{2} \right\} \end{bmatrix}, \tag{5.37}$$

The y-component of $G_s^{\pm}$ is given by

$$\begin{bmatrix} GY^{\pm}(1) \\ GY^{\pm}(2) \\ GY^{\pm}(3) \\ GY^{\pm}(4) \end{bmatrix} = \begin{bmatrix} \rho \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \\ \rho \left\{ \tilde{u}_1 u_2 A_2^{\pm} \pm \tilde{u}_1 B_2 \right\} \\ \rho \left\{ \left(\frac{p}{\rho} + \tilde{u}_2^2 \right) A_2^{\pm} \pm u_2 B_2 \right\} \\ \rho \left\{ \tilde{u}_2 \left(\frac{2\gamma}{\gamma - 1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{A_2^{\pm}}{2} \pm \left(\frac{\gamma + 1}{\gamma - 1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{B_2}{2} \right\} \end{bmatrix}, \tag{5.38}$$

$$
\begin{aligned}
A_1^{\pm} &= \frac{1 \pm \operatorname{erf} S_1}{2}, \quad A_2^{\pm} = \frac{1 \pm \operatorname{erf} S_2}{2}, \\
B_1 &= \frac{e^{-S_1^2}}{2\sqrt{\pi\beta}}, \quad B_2 = \frac{e^{-S_2^2}}{2\sqrt{\pi\beta}}, \quad S_1 = \tilde{u}_1\sqrt{\beta}, \quad S_2 = \tilde{u}_2\sqrt{\beta}, \\
|\tilde{u}| &= \sqrt{\tilde{u}_1^2 + \tilde{u}_2^2} \quad \beta = \frac{1}{2RT}
\end{aligned}
$$

5.3.1 Kinetic treatment of moving boundary condition for solid wall

Consider a boundary point b lying on a moving surface S as shown in fig. 1.3 Let $w_b = w_1 i + w_2 j$ be the grid point velocity along the tangential and the normal direction x-y as shown in the figure. Following the specular reflection principle, the velocity distribution function at b is constructed as the union of two half Maxwellians corresponding to the incident and reflected particles, that is,

$$f_b = F_I \cup F_R \tag{5.39}$$

The incident part corresponds to the particles with the relative velocity normal to the wall $\bar{v_2} < 0$. The reflected part is constructed from the incident part as shown below.

$$
\begin{aligned}
F_I &= F_I(\bar{v_1}, \bar{v_2}) \quad for \quad \bar{v_2} < 0 \\
F_R &= F_I(\bar{v_1}, -\bar{v_2}) \quad for \quad \bar{v_2} > 0
\end{aligned}
$$

The update formula at a boundary point b is then given by

$$\left\langle \Psi, f_b^{n+1} \right\rangle \equiv U_b^{n+1} \equiv \left\langle \Psi, F_I^{n+1} \cup F_R^{n+1} \right\rangle \tag{5.40}$$

where

$$
\begin{aligned}
F_I^{n+1} &= F_I^n - \Delta t \left(\frac{\bar{v_1} + |\bar{v}|}{2} \frac{\partial F}{\partial x} + \frac{\bar{v_1} - |\bar{v}|}{2} \frac{\partial F}{\partial x} + \bar{v_2} \frac{\partial F}{\partial y} \right) \quad for \quad \bar{v_2} < 0 \\
F_R^{n+1} &= F_I^{n+1}(\bar{v_1}, -\bar{v_2}) \quad for \quad \bar{v_2} > 0 \\
U_P^{n+1}(l) &= U_P^n(l) - 2 \ \Delta t \left(\frac{\partial G X_m^{+-}(l)}{\partial X} + \frac{\partial G X_m^{--}(l)}{\partial X} \right. \\
&\qquad \left. + \frac{\partial G Y_m^{+-}(l)}{\partial Y} + \frac{\partial G Y_m^{--}(l)}{\partial Y} \right) \text{ for } l = 1, 2, \&4 \\
&= 0 \quad \text{for } \ l = 3
\end{aligned}
$$

where Δt is the time step, $GX_m^{\pm -}$ and $GY_m^{\pm -}$ represent the x and y quadrant split moving fluxes repectively in the first and second quadrants. For example GX_m^{--}

represents x quadrant split fluxes in the first quadrant, i.e. for $v_1 < 0$, $v_2 < 0$. Similarly $GX_m^{+\,-}$,$GX_m^{+\,+}$ and $GX_m^{-\,+}$ represents x quadrant split fluxes in the second quadrant, third and fourth quadrant respectively.

The moving boundary fluxes which are expressed in terms of the quadrant wise split fluxes are given by the following expressions,

$$G_m^{\pm\,\pm} = [A_b] G_s^{\pm\,\pm} \tag{5.41}$$

where $G_m^{\pm\,\pm}$ represents the quadrant wise split fluxes for a moving grid, $G_s^{\pm\,\pm}$ represents the quadrant wise split fluxes which are similar to that of a static grid except that the velocity components in this case are relative to the grid velocity and matrix $[A_b]$ converts the fluxes to moving grid. The small difference in $[A_b]$ compared to the matrix $[A]$ for interior points is, the element $a_b(3,3)$ of the matrix $[A_b]$ is zero. The transformation matrix $[A_b]$ is given by

$$[A_b] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ w_1 & 1 & 0 & 0 \\ w_2 & 0 & 0 & 0 \\ \frac{w_1^2+w_2^2}{2} & w_1 & w_2 & 1 \end{bmatrix} \tag{5.42}$$

The x-component of the quadrant split fluxes is given by,

$$\begin{bmatrix} GX(1)^{\pm\,\pm} \\ GX(2)^{\pm\,\pm} \\ GX(3)^{\pm\,\pm} \\ GX(4)^{\pm\,\pm} \end{bmatrix} = \begin{bmatrix} \rho A_2^{\pm} \left\{ \tilde{u}_1 A_1^{\pm} \pm B_1 \right\} \\ \rho A_2^{\pm} \left\{ \left(\frac{p}{\rho} + \tilde{u}_1^2 \right) A_1^{\pm} \pm \tilde{u}_1 B_1 \right\} \\ \rho \left\{ \tilde{u}_1 A_1^{\pm} \pm B_1 \right\} \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \\ \rho A_2^{\pm} \left\{ \tilde{u}_1 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{A_1^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{B_1}{2} \right\} \pm \frac{1}{2} \tilde{u}_2 B_2 \left\{ \tilde{u}_1 A_1^{\pm} \pm B_1 \right\} \end{bmatrix}. \tag{5.43}$$

The y-component of the quadrant split fluxes are given by,

$$\begin{bmatrix} GY(1)^{\pm\,\pm} \\ GY(2)^{\pm\,\pm} \\ GY(3)^{\pm\,\pm} \\ GY(4)^{\pm\,\pm} \end{bmatrix} = \begin{bmatrix} \rho A_2^{\pm} \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \\ \rho A_2^{\pm} \left\{ \left(\frac{p}{\rho} + \tilde{u}_2^2 \right) A_2^{\pm} \pm \tilde{u}_2 B_2 \right\} \\ \rho \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \\ \rho A_2^{\pm} \left\{ \tilde{u}_2 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{A_2^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + |\tilde{u}|^2 \right) \frac{B_2}{2} \right\} \pm \frac{1}{2} \tilde{u}_2 B_2 \left\{ \tilde{u}_2 A_2^{\pm} \pm B_2 \right\} \end{bmatrix}. \tag{5.44}$$

5.3.2 Kinetic treatment of moving boundary condition for outer boundary

The treatment of the boundary condition for moving outer boundary [41] is very much similar to the above analysis, which in turn is very much similar to the kinetic outer boundary condition developed by Ramesh et al.[66] for stationary grids. As before, the velocity distribution at any point b on the moving outer boundary is constructed as the union of two half Maxwellians corresponding to incoming and outgoing particles. The velocity distribution corresponding to the incoming particles is constructed from the freestream conditions and that for the outgoing particles is obtained from the interior points. Therefore for the union of the two velocity distributions we have,

$$f_b^{n+1} = F_{in}^{n+1} \cup F_{out}^{n+1} \tag{5.45}$$

where

$$\begin{aligned} F_{in}^{n+1} &= F_{\infty}^{n+1} \quad for \quad \bar{v_2} < 0 \\ &= \frac{\rho_\infty}{I_{0\infty}} \frac{\beta_\infty}{\pi} \exp\left[-\beta(v_1 - u_{1\infty})^2 - \beta(v_2 - u_{2\infty})^2 - I/I_0\right] \\ F_{out}^{n+1} &= F_{out}^{n} - \Delta t \left(\frac{\bar{v_1} + |\bar{v_1}|}{2} \frac{\partial F}{\partial x} + \frac{\bar{v_1} - |\bar{v_1}|}{2} \frac{\partial F}{\partial x} + \bar{v_2} \frac{\partial F}{\partial y} \right) \quad for \quad \bar{v_2} > 0 \end{aligned}$$

Taking Ψ moments of the eq. (5.45) we get the update formula for the outer boundary points as,

$$U_b^{n+1} = \overline{U}_m^n - \Delta t \left(\frac{\partial GX_m^{+\,+}}{\partial X} + \frac{\partial GX_m^{-\,+}}{\partial X} + \frac{\partial GY_m^{+\,+}}{\partial Y} + \frac{\partial GY_m^{-\,+}}{\partial Y} \right) \tag{5.46}$$

where Δt is the time step, $GX_m^{\pm\,+}$ and $GY_m^{\pm\,+}$ represent the quadrant split moving fluxes in the third and fourth quadrants respectively. The expressions are already given in previous section. The term $\overline{U}_m^n$ in eq. (5.46)is function of U_∞ and U_b^n given by the expression,

$$\begin{bmatrix} \overline{U(1)} \\ \overline{U(2)} \\ \overline{U(3)} \\ \overline{U(4)} \end{bmatrix} = \begin{bmatrix} \left\{\rho A_2^-\right\}_\infty + \left\{\rho A_2^+\right\}_b \\ \left\{\rho \tilde{u}_1 A_2^-\right\}_\infty + \left\{\rho \tilde{u}_1 A_2^+\right\}_b \\ \left\{\rho\left(\tilde{u}_2 A_2^- - B_2\right)\right\}_\infty + \left\{\rho\left(\tilde{u}_2 A_2^+ + B_2\right)\right\}_b \\ \left\{\rho A_2^- e - \frac{1}{2}\rho \tilde{u}_2 B_2\right\}_\infty + \left\{\rho A_2^+ e + \frac{1}{2}\rho \tilde{u}_2 B_2\right\}_b \end{bmatrix} \tag{5.47}$$

where subscript ∞ refers to freestream condition and $e = \frac{p}{\rho(\gamma-1)} + \frac{1}{2}(\tilde{u}_1^2 + \tilde{u}_2^2)$.

5.3.3 Results and Discussions for 2-D problem

The present method, that is, LSKUM on moving grid has been applied to computation of unsteady flow past an airfoil undergoing pitching oscillations[42, 69]. The airfoil chosen is NACA 0012. The free stream mach number for this case is $M_\infty = 0.755$. This is a standard AGARD[3] test case 1(CT1) and has been used by many investigators for checking their numerical algorithms. The oscillation cycle is defined by,

$$\alpha = \alpha_m + \alpha_0 sin(\omega t) \quad \text{where} \quad \alpha_m = 0.016^0 \quad \alpha_0 = 2.51^0.$$

Reduced frequency based on chord length c of the airfoil is $k = \omega c/2U_\infty = 0.0814$, where ω is the circular pitch frequency and U_∞ is free stream fluid velocity. This case corresponds to AGARD Test case no. 1(CT1). Point distribution from an unstructured grid with 4074 points has been used. It has 160 points on the airfoil and 40 points on the farfield boundary. Farfield boundary is at 10 chords distance. The necessary connectivity information was generated using quad-tree based search algorithm. There are many ways of moving the grid. One simple method employed in the present work is pitching up and down of the whole grid along with the airfoil. For such a grid movement it is very easy to calculate the grid velocity. Figures 5.12 and 5.13 show the comparison of computations with experiment of AGARD values, the lift coefficient C_l and moment coefficient C_m versus instantaneous angle of attack α is considered for such a comparison. Figures 5.14 to 5.15 show instantaneous pressure contours for various angles of attack clearly showing the movement of the shock at different time instants characterising the unsteady behaviour of the flow. Figures 5.22to 5.29 show instantaneous C_p plots for various angles of attack.

5.3.4 Conclusions

The Least squares Kinetic Upwind method has been extended in a novel way to problems involving moving grid. This method called LSKUM-MG(LSKUM on Moving Grid) is based on introduction of special derivative following a grid point. The LSKUM-MG formulation reduces to KSPH[48] method when the grid point velocity becomes equal to the fluid velocity. Thus KSPH method can be considered as a special case of LSKUM-MG. The present LSKUM-MG solver has been applied to the 1-D moving piston problem and accurate results have been obtained. Also the 2-D LSKUM-MG has been used succesfully to compute the unsteady flow past airfoil oscillating in pitch. Treatment of boundary conditions on moving walls has also been developed within the kinetic framework. The computed results compare quite well with the experimental results of the AGARD[3] test case.

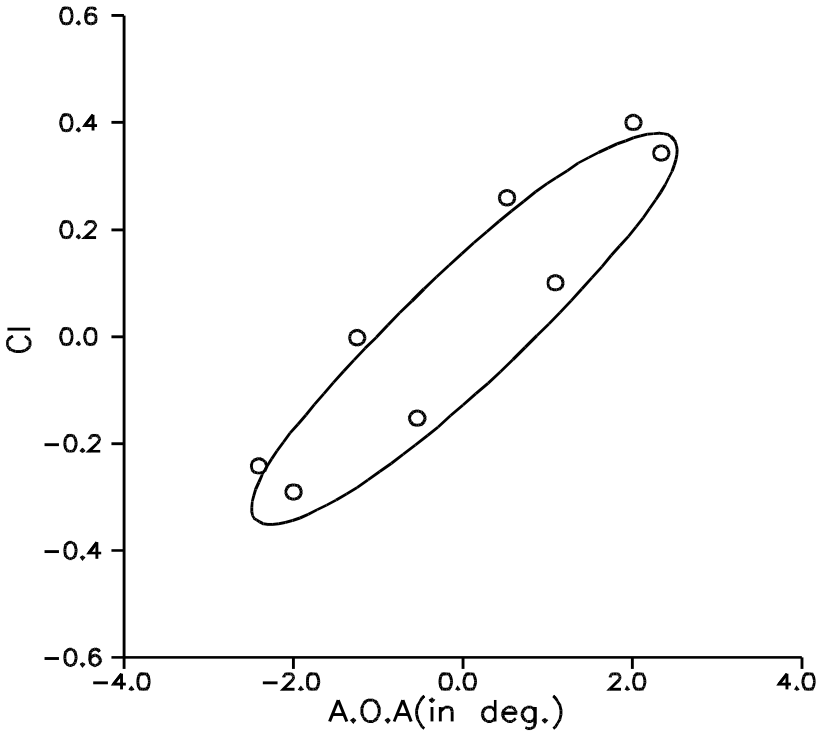


Figure 5.12: Hysteris loop for lift coefficient

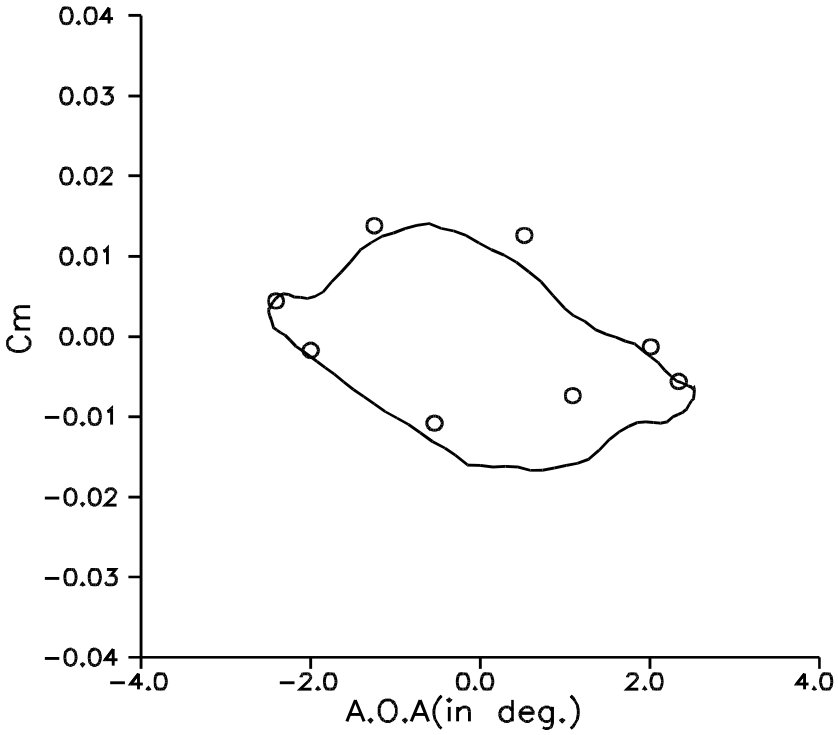


Figure 5.13: Hysteris loop for moment coefficient

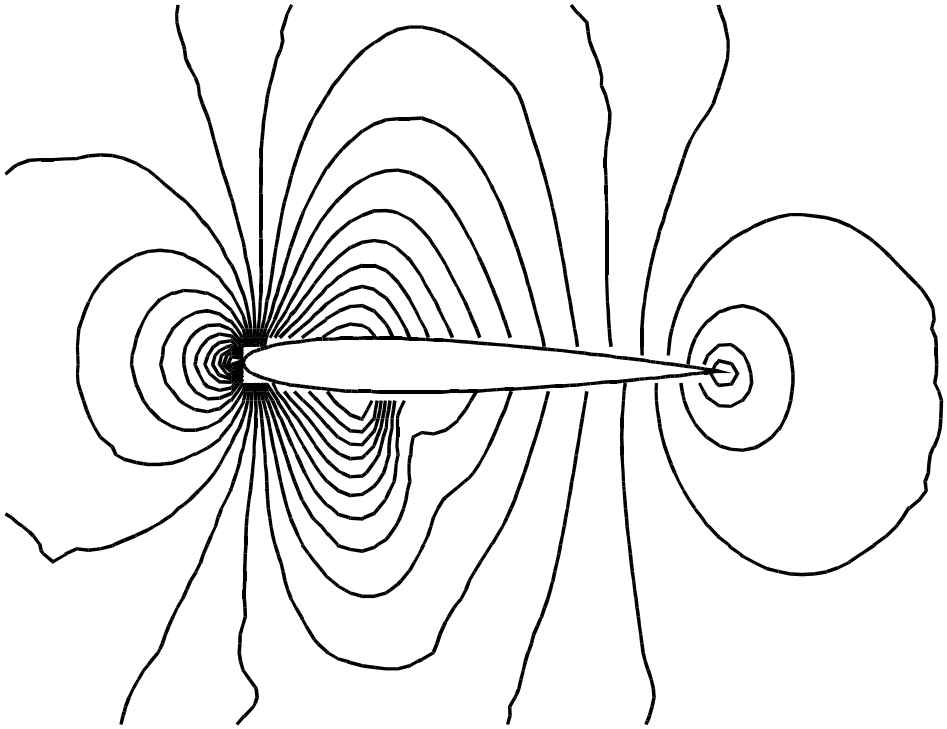


Figure 5.14: Pressure contours : AOA=1.09^o (up)

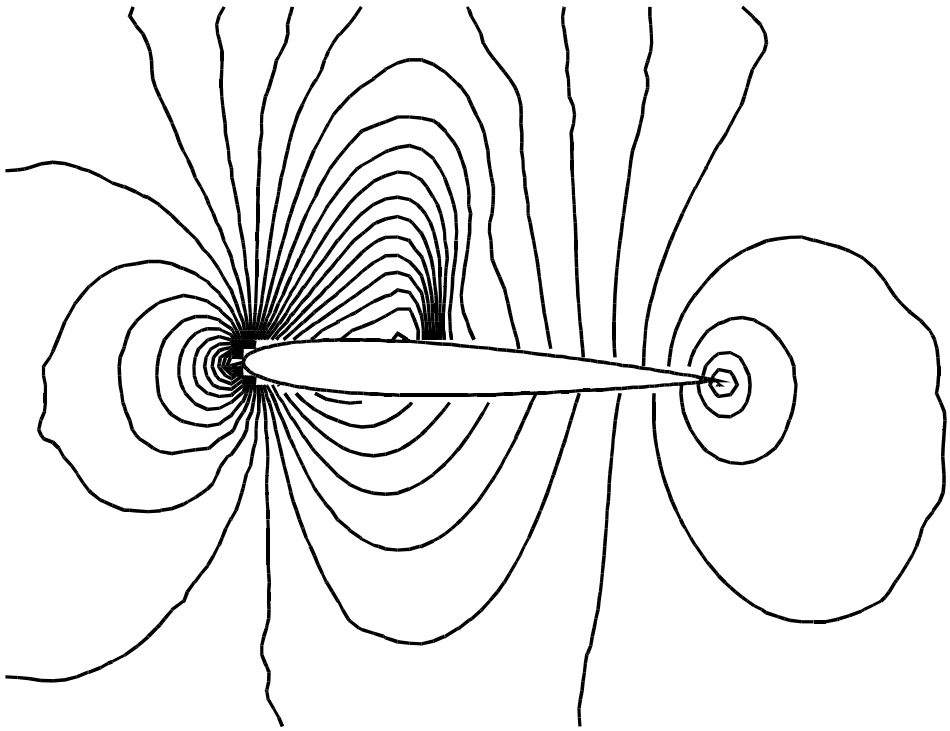


Figure 5.15: Pressure contours : AOA=2.34^o (down)

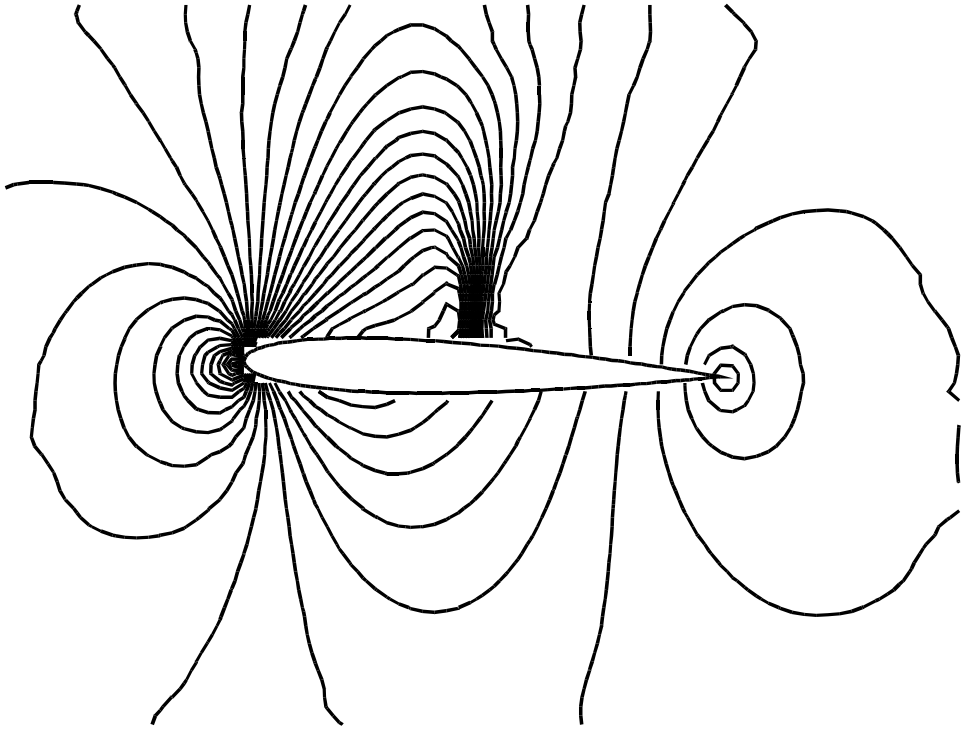


Figure 5.16: Pressure contours : AOA=2.01^o (down)

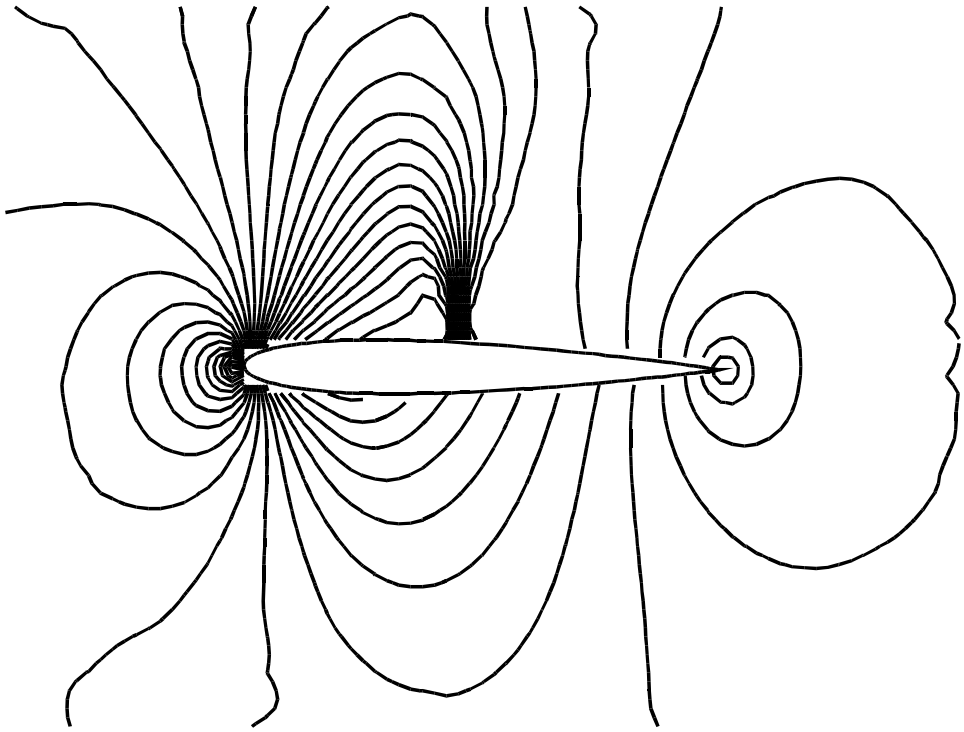


Figure 5.17: Pressure contours : AOA=0.52^o (down)

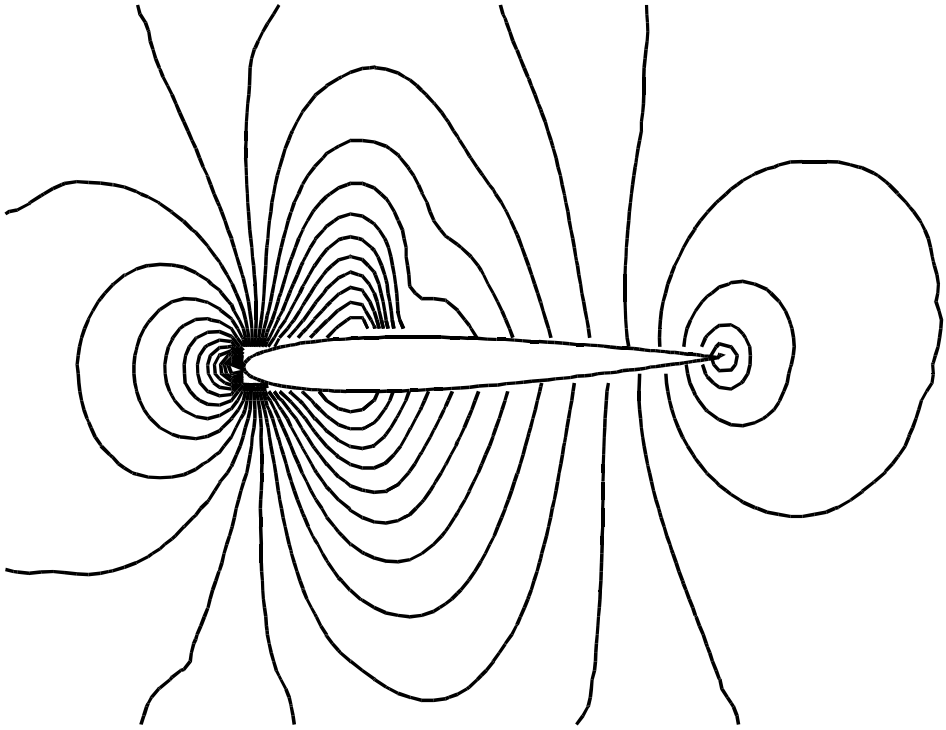


Figure 5.18: Pressure contours : AOA=-1.25^o (down)

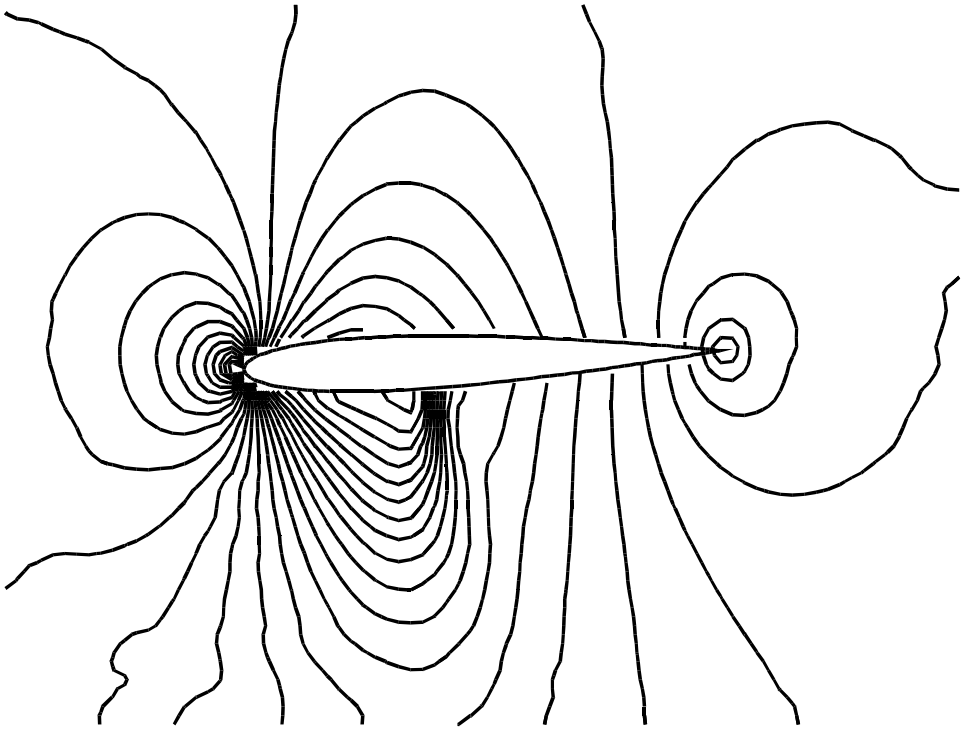


Figure 5.19: Pressure contours : AOA=-2.41^o (down)

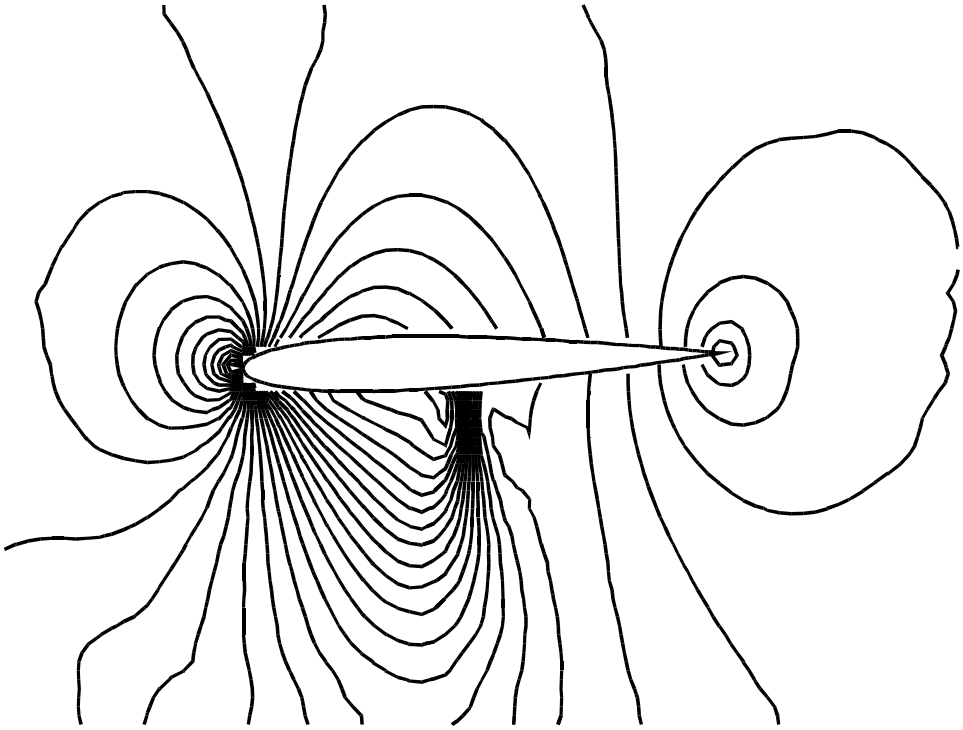


Figure 5.20: Pressure contours : AOA$=-2.0^o$ (up)

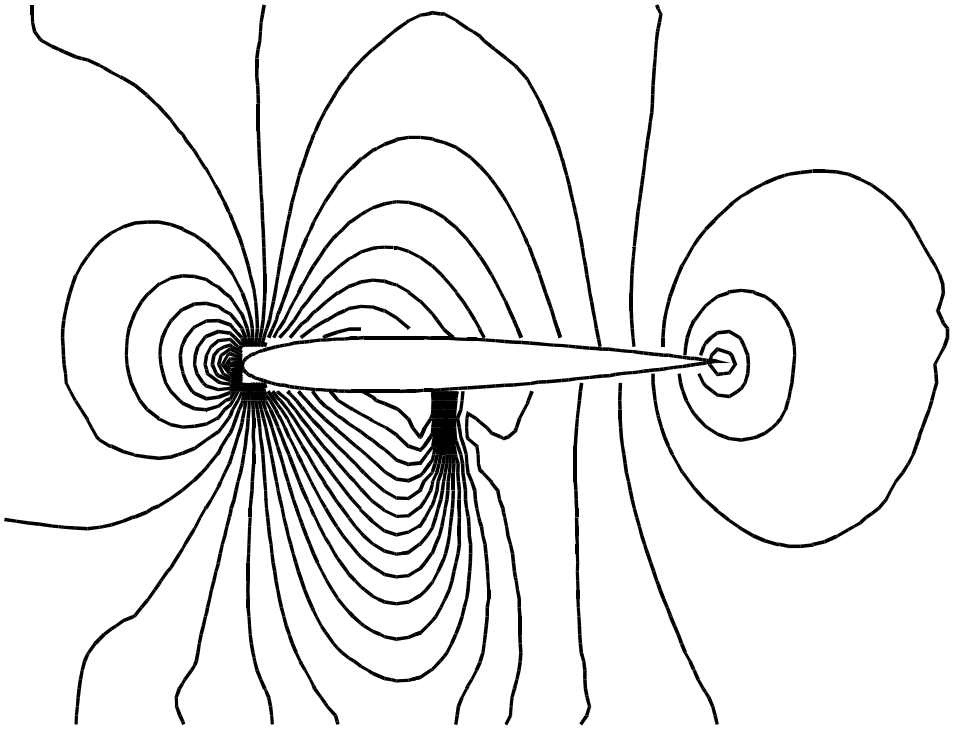


Figure 5.21: Pressure contours : AOA$=-0.54^o$ (up)

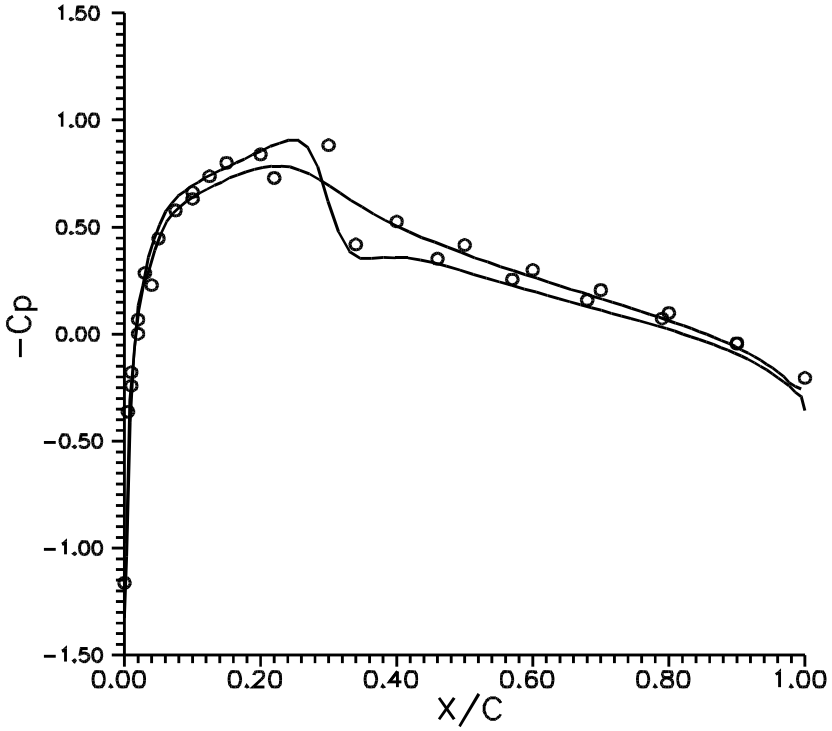


Figure 5.22: Cp plots : AOA=1.09^o (up)

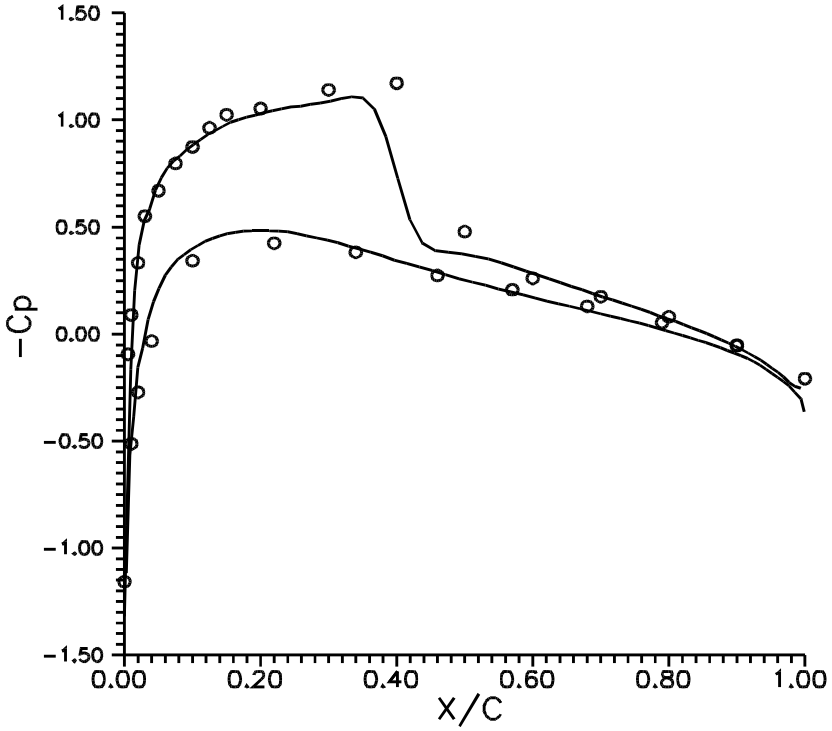


Figure 5.23: Cp plots : AOA=2.34^o (down)

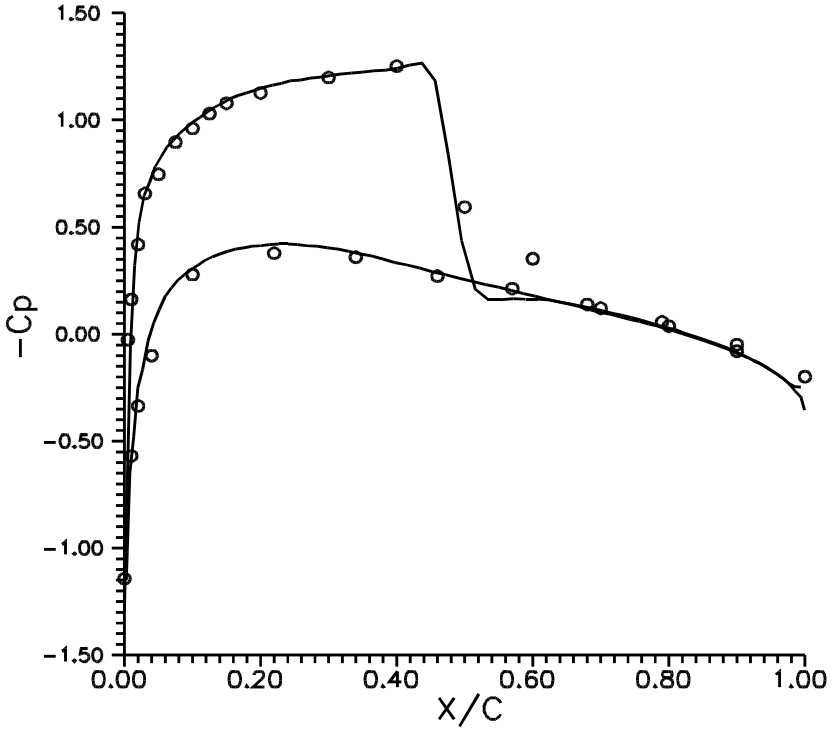


Figure 5.24: Cp plots : AOA=2.01^o (down)

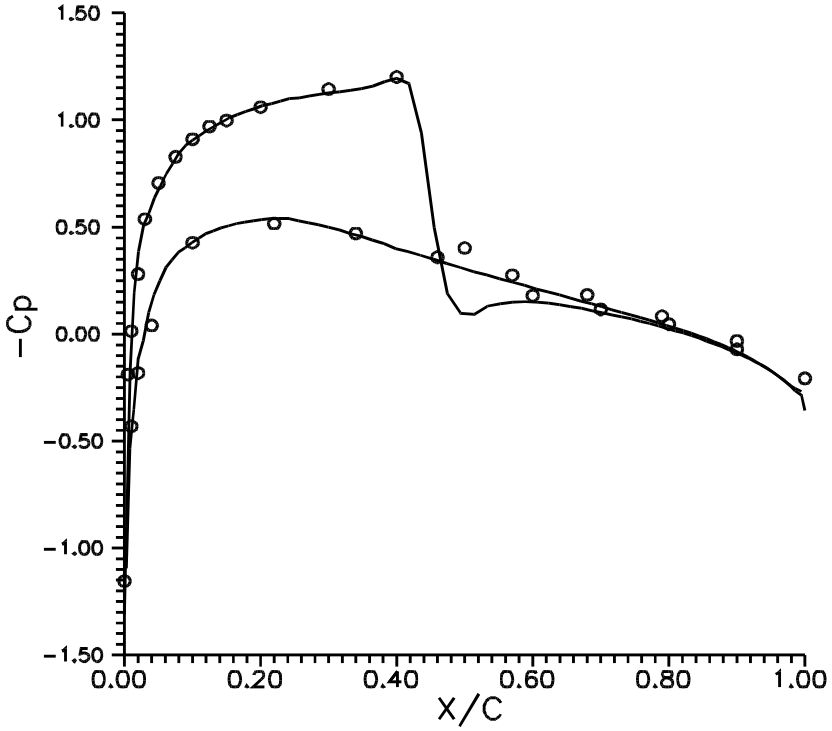


Figure 5.25: Cp plots : AOA=0.52^o (down)

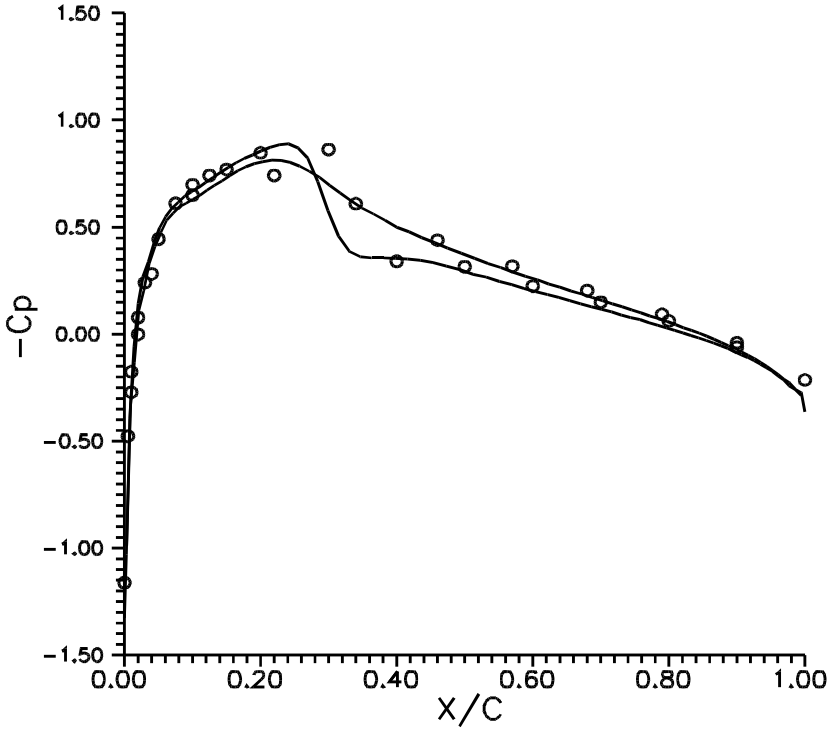


Figure 5.26: Cp plots : AOA=-1.25^o (down)

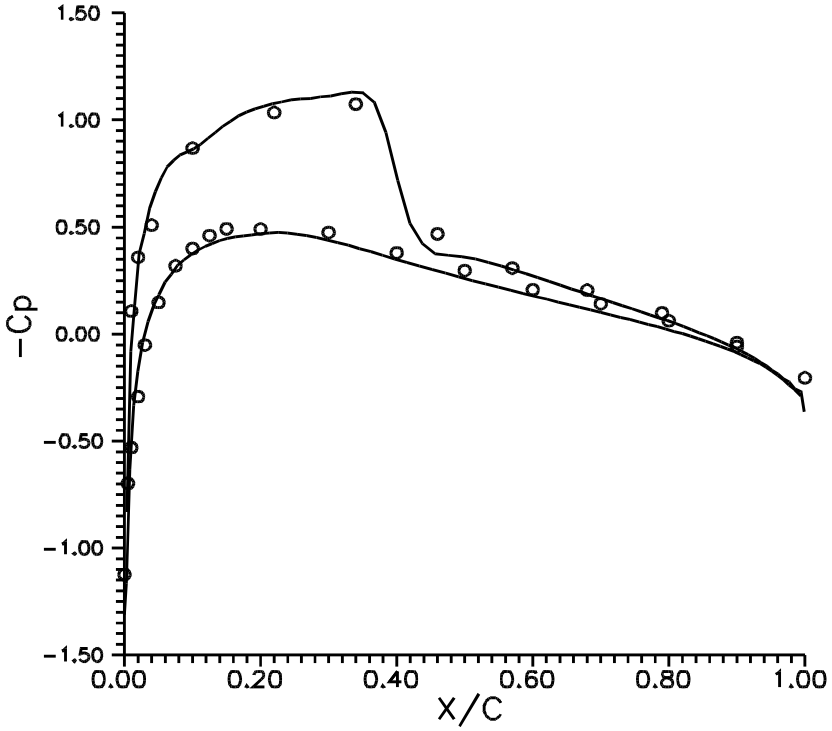


Figure 5.27: Cp plots : AOA=-2.41^o (down)

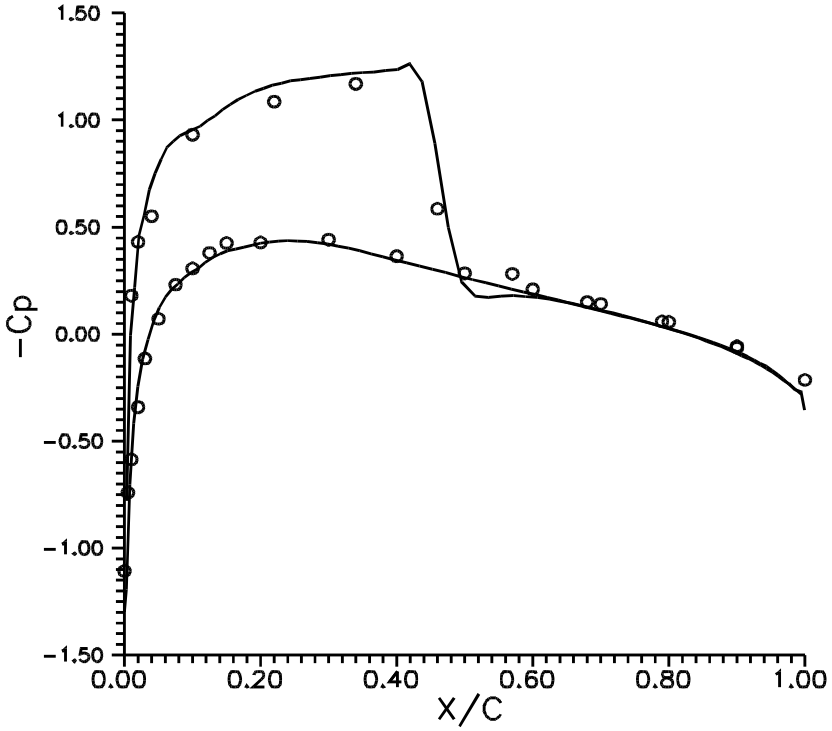


Figure 5.28: Cp plots : AOA=-2.0^o (up)

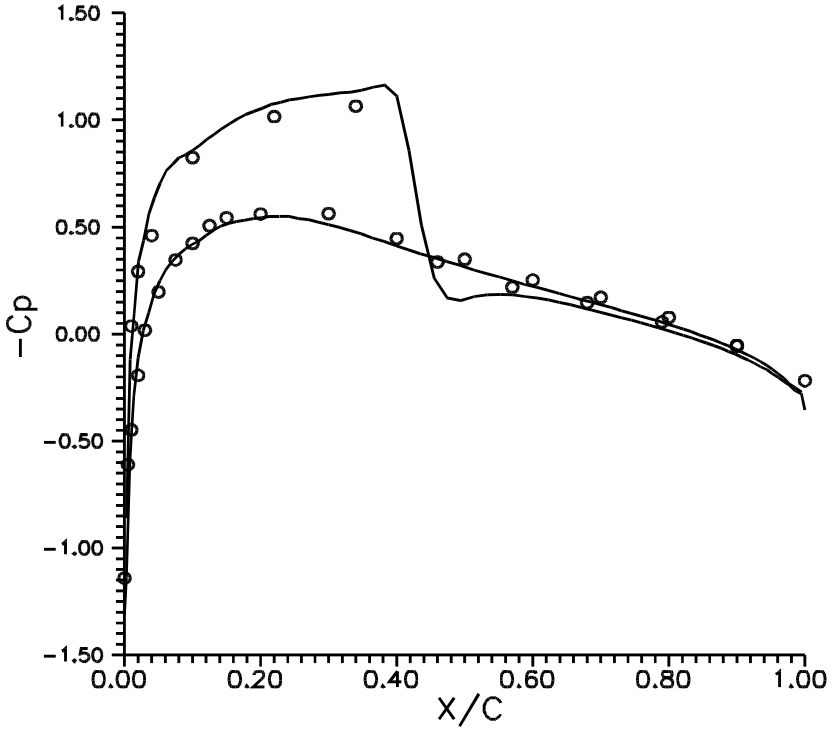


Figure 5.29: Cp plots : AOA=-0.54^o (up)

5.4 Novel Higher order LSKUM(LSKUM-NH)

LSKUM has been shown to work on a variety of grids for different types of flows. Presently lot of new schemes are being developed towards Computational Aeroacoustic Applications(CAA), which basically require higher order accurate methods with minimum dissipation and dispersion. The motivation for the present work was to make an attempt to modify the present LSKUM for obtaining progressively higher order derivatives within the least squares framework . In order to achieve progressively higher order accuracy, a new version of LSKUM has been developed. In the existing method for example, second order calculation at any point in the field requires the data of first order derivatives at its secondary nodes(very similar to the arguments we presented in section 5.2.1). This in turn would depend upon the data of the field variables at its tertiary points. Thus in effect the connectivity set gets enhanced. This becomes further more expanded when we need further higher order schemes. But from the point of accuracy it would be desirable not to have influence of more and more farther points. In order to have higher order LSKUM but yet the stencil of supporting points remaining same, has led to the development of the novel higher order LSKUM operating on compact stencils. This is called as the LSKUM-NH solver.

5.4.1 Higher order accurate LSKUM without stencil augmentation

Consider the 1-D Boltzmann equation

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} = J \tag{5.48}$$

where f is the velocity distribution function, v is the molecular velocity and x is the molecular position. J represents a collision term which vanishes in the Euler limit, when f is a Maxwellian distribution. The Maxwellian distribution, F, in one dimension is given by

$$F = \frac{\rho}{I_0}(\frac{\beta}{\pi})^{\frac{1}{2}} \exp\left[-\beta(v-u)^2 - I/I_0\right] \tag{5.49}$$

where v is the molecular velocity, u is the fluid velocity, $\beta = 1/(2RT)$, and I_0 is the internal energy due to non-translational degrees of freedom, $I_0 = \frac{2-3\gamma}{2(\gamma-1)}RT$, I is the internal energy variable, T is the fluid temperature and R gas constant.

Therefore in the Euler limit we get

$$\frac{\partial F}{\partial t} + v \cdot \frac{\partial F}{\partial x} = 0 \tag{5.50}$$

Using single step Euler for time discretization, splitting the molecular velocity into positive and negative parts we can write the update for F as

$$F^{n+1} = F^n - \left(\frac{v + |v|}{2} \cdot F_x + \frac{v - |v|}{2} \cdot F_x \right) \tag{5.51}$$

The derivative F_x is evaluated using the Least Squares approach. Consider any point o as shown in Figure 5.2. F_{x_o} is evaluated using the data of F at the surrounding points. The least squares formula for F_{x_o} is given by

$$F_{x_o} = \frac{\sum_{i=1}^{N} \Delta x_i \Delta F_i}{\sum_{i=1}^{N} \Delta {x_i}^2} \tag{5.52}$$

where $\Delta x_i = x_i - x_o$, $\Delta F_i = F_i - F_o$ and N represents the total number of points in the neighbourhood. For $v > 0$, all the points in stencil are chosen to left of point o and for $v < 0$, points are chosen to the right of o. Second order accuracy is achieved by two step defect correction method [31]. In step one the first order approximation to F_x is obtained by using equation (5.52). In step two ΔF_i is modified as

$$\Delta \tilde{F}_i = \Delta F_i - \frac{\Delta x_i \Delta F_{x_i}}{2} \tag{5.53}$$

where $\Delta F_{x_i} = F^1_{x_i} - F^1_{x_o}$ and superscript 1 indicates that the derivative used is first order approximate. With this modified difference, equation (5.52) is used to get the second order approximation for F_{x_o}. From these discussions we can clearly see that in order to evaluate $F^2_{x_o}$(second order accurate F_{x_o}), we need the data of the $F^1_{x_i}$ at the surrounding points. This in turn implies that data of F is further needed at the neighbouring points of the neighbouring points. Thus in effect the stencil gets enlarged for the evaluation of $F^2_{x_o}$. This will progressively increase with the increase in the need for higher order accurate F_x. In the next section we describe the new approach, where higher order accurate F_x can be evaluated with the stencil size remaining essentially the same.

Consider the weighted Least Squares formula

$$F_{x_o} = \frac{\sum_{i=1}^{N} \Delta x_i \Delta F_i w_i}{\sum_{i=1}^{N} \Delta {x_i}^2 w_i} \tag{5.54}$$

where w_i is the weight associated with each point. The weighting function used is usually of the form $w_i = \frac{1.}{d_i^p}$ where d_i is the distance between any point i in the connectivity with the point o.

Expanding ΔF_i using Taylor series we have,

$$\Delta F_i = \Delta x_i F_{x_e} + \frac{\Delta x_i^2}{2} F_{xx_e} + \frac{\Delta x_i^3}{6} F_{xxx_e} + h.o.t. \tag{5.55}$$

The subscript e denotes exact values. Substituting the above equation in equation (5.54) we get

$$F^1_x = F_{x_e} + \frac{\sum \frac{\Delta x_i^3 w_i}{2}}{\sum \Delta x_i^2 w_i} F_{xx_e} + \frac{\sum \frac{\Delta x_i^4 w_i}{6}}{\sum \Delta x_i^2 w_i} F_{xxx_e} + h.o.t. \tag{5.56}$$

Let $a(w) = \frac{\sum \frac{\Delta x_i^3 w_i}{2}}{\sum \Delta x_i^2 w_i}$ and $b(w) = \frac{\sum \frac{\Delta x_i^4 w_i}{6}}{\sum \Delta x_i^2 w_i}$. Above equation can be written as

$$F_x^1(w) = F_{x_e} + a(w)F_{xx_e} + b(w)F_{xxx_e} + h.o.t. \tag{5.57}$$

The basic idea for obtaining higher order accurate F_x is as follows. We use two different weighting functions $w1$ and $w2$ and get two first order estimates for the derivative. This can be written as follows.

$$F_x^1(w1) = F_{x_e} + a(w1)F_{xx_e} + b(w1)F_{xxx_e} + h.o.t.$$
$$F_x^1(w2) = F_{x_e} + a(w2)F_{xx_e} + b(w2)F_{xxx_e} + h.o.t.$$

Now we can easily see that using the above two equations we can eliminate the F_{xx_e} term to get the second order estimate to F_x as

$$F_x^2 = \left(\frac{a(w2)}{a(w2) - a(w1)}F_x^1(w1) - \frac{a(w1)}{a(w2) - a(w1)}F_x^1(w2)\right) + \frac{a(w2)b(w1) - a(w1)b(w2)}{a(w2) - a(w1)}F_{xxx_e} + h.o.t.$$

Thus we have

$$F_x^2 = \alpha F_x^1(w1) + \beta F_x^1(w2) + O(\Delta x^2)F_{xxx_e} \tag{5.58}$$

where $\alpha = \frac{a(w2)}{a(w2)-a(w1)}$, $\beta = -\frac{a(w1)}{a(w2)-a(w1)}$ and $\alpha + \beta = 1$. From the above formula we can see that this way we are able to get the second order approximation for F_x^2 without the need of F_x^1 at the surrounding points. Thus in effect we are able to get the second order estimate with the same stencil as needed for the first order evaluaution.

The above procedure can be easily extended for further higher order schemes. As an illustration, we give here some details for a third order LSKUM-NH.

Let us choose three weighting functions $w1, w2$ and $w3$. Using these three weighting functions and eq. (5.57) we can write three first order estimates for F_x as

$$F_x^1(w1) = F_{x_e} + a(w1)F_{xx_e} + b(w1)F_{xxx_e} + h.o.t.$$
$$F_x^1(w2) = F_{x_e} + a(w2)F_{xx_e} + b(w2)F_{xxx_e} + h.o.t.$$
$$F_x^1(w3) = F_{x_e} + a(w3)F_{xx_e} + b(w3)F_{xxx_e} + h.o.t.$$

From the above three eqs., neglecting the higher order terms we can easily solve for a third order estimate for F_x.

5.4.2 Results and discussions

We have applied this new LSKUM-NH for a shock tube problem. Figures 5.30 shows a comparison of the density propagation using first order, two step second order and the new second order LSKUM-NH for the shock tube problem. It can be observed that the new method compares very well with the exact solution. Also the plot suggests that the new method predicts slightly more sharper shocks. In the present computations

the two weight functions we have chosen for the LSKUM-NH calculations are $W_1 = \frac{1}{d^6}$ and $W_2 = \frac{1}{d^0}$. We can notice that W_2 is the uniform weight function. Figures 5.31,5.32 and 5.33 show the pressure, temperature and velocity plots. One can see from these figures that the new method perfoms quite well, also the velocity plot shows that all the variations are well predicted without any oscillations in the solution. Further testing of the method in two dimensions is neccessary to fully establish this approach. We feel that these preliminary results are quite encouraging and have good future for extension to higher dimensions.

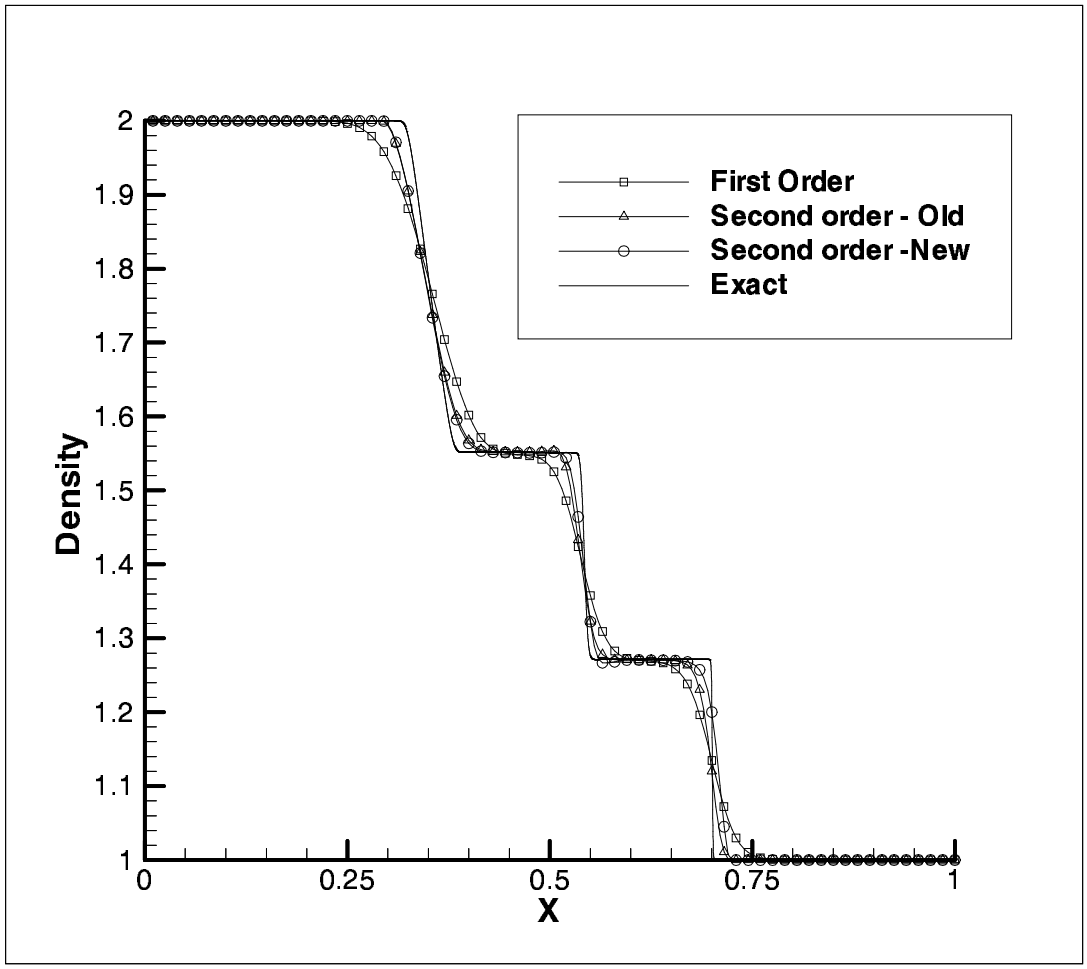


Figure 5.30: Density plot : 1-D shock tube problem : LSKUM-NH

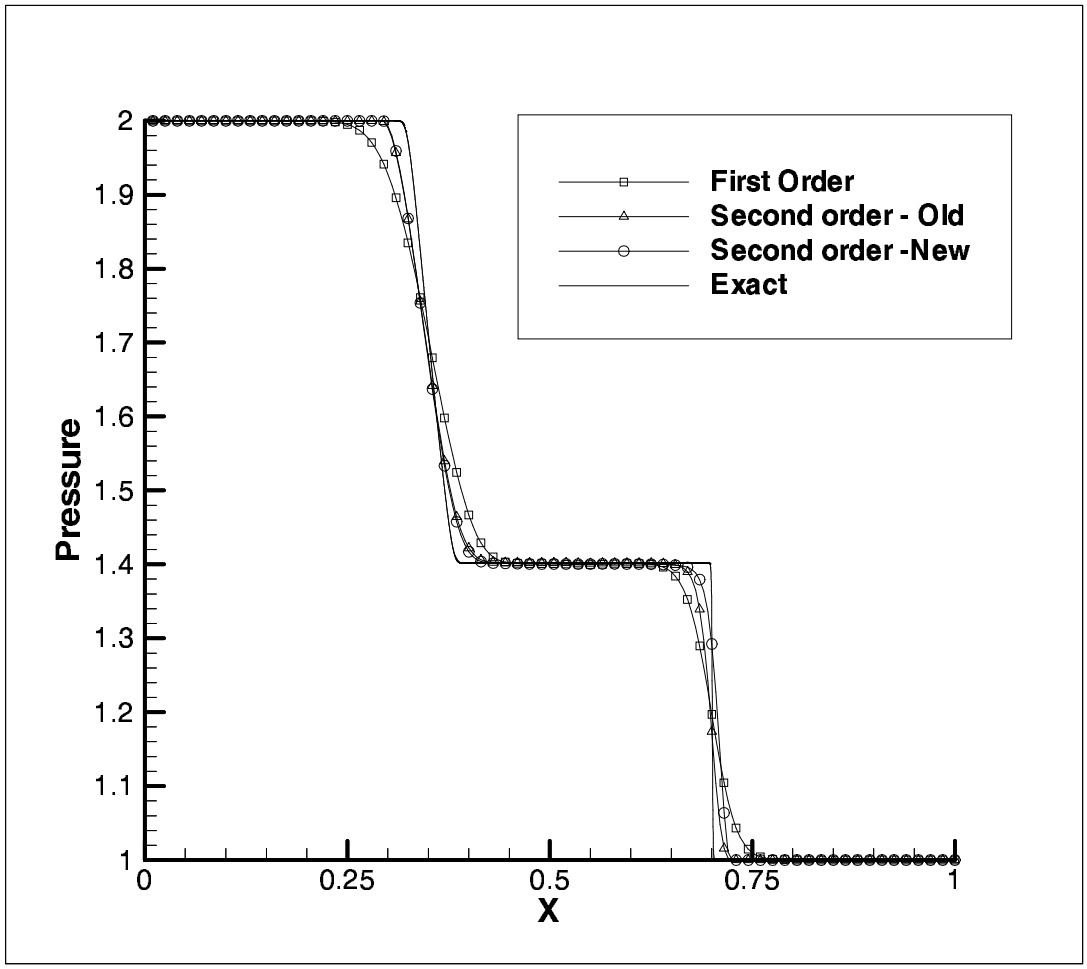


Figure 5.31: Pressure plot : 1-D shock tube problem : LSKUM-NH

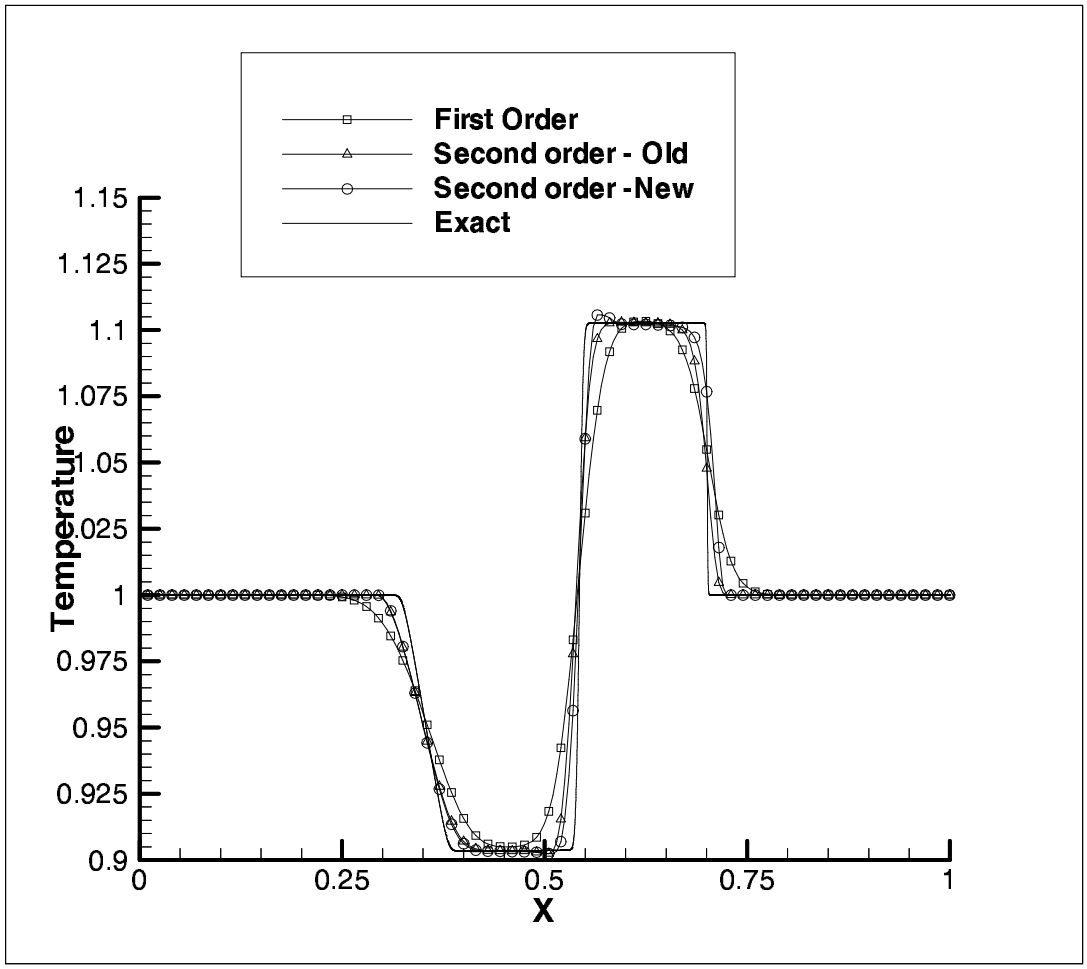


Figure 5.32: Temperature plot : 1-D shock tube problem : LSKUM-NH

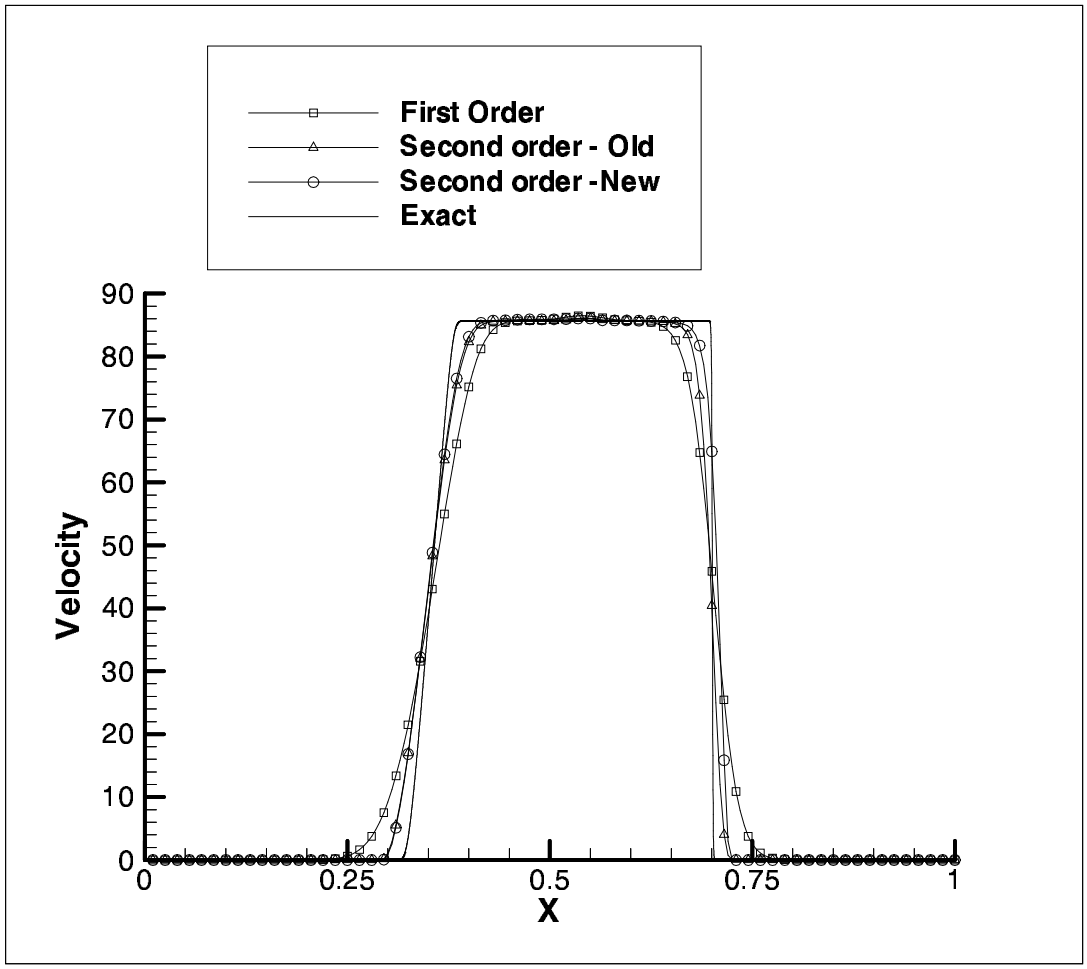


Figure 5.33: Velocity plot : 1-D shock tube problem : LSKUM-NH

Chapter 6

Concluding remarks

The grid free Least Squares Kinetic Upwind Method(LSKUM) is a node based solver capable of operating on any type of distribution of points. It does not need any tesselation of points to form edges or rays as needed by finite volume methods. What is needed is the neighbourhood information of each node called as connectivity data for the given distribution of points. For the purpose of obtaining connectivity data we have developed a quadtree based preprocessor which generates the connectivity information for any arbitrary distribution of points. In order for the solver to operate satisfactorily, connectivity data should have certain desirable properties. These are, certain minimum number of points in the connectivity data, non occurrence of degeneracy due to all points in the connectivity lying on a straight line, connectivity also should contain only points which are aerodynamically related. Hence in our present work, we have built in certain logic in the preprocessor so that the connectivity data generated satisfies all these requirements. The preprocessor along with the LSKUM solver has been shown to work on a variety of distribution of points. A new boundary condition based on kinetic theory called as Kinetic Outer Boundary condition(KOBC) has been developed and incorporated for all 2-D and 3-D calculations. We have also shown that KOBC has only a single boundary treatment irrespective of the flow conditions at the outer boundary. Along with this new boundary condition and in conjunction with the preprocessor we extensively demonstrated the capability of the solver to operate for subsonic, transonic as well as supersonic flows on a variety of distribution of points. We have also extended the LSKUM method to 3-D flows. The formulation of 3-D LSKUM along with boundary treatment for solid walls as well as the far field has been presented. A code based on 3-D LSKUM called as Super-BHEEMA has been written. We have demonstrated the satisfactory performance of the Super-BHEEMA for different test cases. Supersonic flow past a hemisphere has been solved satisfactorily. We have also used the 3-D Super-BHEEMA code to simulate a 3-D intense spherically symmetric blast wave in air. An interesting feature of this test case is, we have used a Cartesian grid to simulate the blast wave which is spherically symmetric. Even though the flow conditions are very extreme in nature we have been able to compute this flow and also capture the spherical symmetry

on a Cartesian grid. 3-D LSKUM solver was used to compute transonic flow past ONERA-M6 wing which is a standard AGARD test case. For this case also we got results comparing very favourably with the experiment. Finally the Super-BHEEMA code was used in computation of Hypersonic flow past a complex generic flight vehicle configuration. The computed aerodynamic coefficients compared favourably with the available experimental data.

Having demonstrated the capability of 2-D as well as 3-D LSKUM solver for a range of applications, finally we have also presented a few additional novel developmets of LSKUM. We have presented a new formulation for the LSKUM on Moving Grids(LSKUM-MG). Again one of the very attractive features of this formulation is that, it has has the ability to operate on any arbitrary mesh having arbitrary grid point velocities. In the present work we have also applied LSKUM-MG to compute a few 1-D as well as 2-D test cases. We have shown that 1-D LSKUM-MG simulates the moving piston problem very accurately. For the 2-D test case we have demonstrated the method by computing the flow past an oscillating NACA 0012 airfoil. This is a standard AGARD test case. For this test case also we have obtained very good results. In general moving grids involve moving boundary conditions. In the present work for LSKUM-MG we have also presented the new extensions of the boundary conditions for the moving grid. These include the moving boundary condition for the solid wall and the moving boundary condition for the far field boundary. In the end we have presented a new approach for obtaining higher order accurate schemes for the Least Squares method. This new method called as Novel Higher order LSKUM(LSKUM_NH) has also been applied for 1-D Euler Equations. We have demonstrated the above method for a shock propogation problem. Finally we conclude the thesis by summarising the main contributions :

- Formulation for 3-D Least Squares Kinetic Upwind Method.
- A new kinetic theory based approach for the treatment of far field boundary condition in 2-D and 3-D
- Development of a general purpose preprocessor based on quadtree data structures for the connectivity generation for 2D problems.
- Application of 2-D LSKUM in conjunction with the quadtree preprocessor on a variety of point distributions.
- Development and application of 3-D LSKUM solver for several test cases.
- Additional novel developments on LSKUM.
    1. Extension of LSKUM to moving grids.
    2. Application of LSKUM-MG to 1-D and 2-D problems
    3. Development of a new novel higher order method for LSKUM(LSKUM-NH) and its application to 1-D shock tube problem.

Appendix A

Split Fluxes 2-D LSKUM

The expressions for the split fluxes, in terms of the flow variables, are given below. The x-component of the flux is given by

$$\begin{bmatrix} GX^{\pm}(1) \\ GX^{\pm}(2) \\ GX^{\pm}(3) \\ GX^{\pm}(4) \end{bmatrix} = \begin{bmatrix} \rho\left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho\left\{\left(\frac{p}{\rho} + u_1^2\right) A_1^{\pm} \pm u_1 B_1\right\} \\ \rho\left\{u_1 u_2 A_1^{\pm} \pm u_2 B_1\right\} \\ \rho\left\{u_1\left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho} + u_1^2 + u_2^2\right)\frac{A_1^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho} + u_1^2 + u_2^2\right)\frac{B_1}{2}\right\} \end{bmatrix}, \quad \text{(A.1)}$$

The y-component of the flux is given by

$$\begin{bmatrix} GY^{\pm}(1) \\ GY^{\pm}(2) \\ GY^{\pm}(3) \\ GY^{\pm}(4) \end{bmatrix} = \begin{bmatrix} \rho\left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho\left\{u_1 u_2 A_2^{\pm} \pm u_1 B_2\right\} \\ \rho\left\{\left(\frac{p}{\rho} + u_2^2\right) A_2^{\pm} \pm u_2 B_2\right\} \\ \rho\left\{u_2\left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho} + u_1^2 + u_2^2\right)\frac{A_2^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho} + u_1^2 + u_2^2\right)\frac{B_2}{2}\right\} \end{bmatrix}, \quad \text{(A.2)}$$

A.1 Quadrantwise Split Fluxes for the KFVS-LSKUM

In case of quadrant splitting GX and GY split to four components each. They are denoted by GX^I, GX^{II}, GX^{III}, GX^{IV}, GY^I, GY^{II}, GY^{III}, GY^{IV}. The x-component

of the split fluxes in the first quadrant are

$$\begin{bmatrix} GX^I(1) \\ GX^I(2) \\ GX^I(3) \\ GX^I(4) \end{bmatrix} = \begin{bmatrix} \rho A_2^- \left\{ u_1 A_1^- - B_1 \right\} \\ \rho A_2^- \left\{ \left(\frac{p}{\rho} + u_1^2 \right) A_1^- - u_1 B_1 \right\} \\ \rho \left\{ \left(u_1 A_1^- - B_1 \right) \left(u_2 A_2^- - B_2 \right) \right\} \\ \rho A_2^- \left\{ u_1 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_1^-}{2} - \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_1}{2} \right\} \\ -\rho u_2 \frac{B_2}{2} \left(u_1 A_1^- - B_1 \right) \end{bmatrix}, \quad \text{(A.3)}$$

The x-component of the split fluxes in the second quadrant are

$$\begin{bmatrix} GX^{II}(1) \\ GX^{II}(2) \\ GX^{II}(3) \\ GX^{II}(4) \end{bmatrix} = \begin{bmatrix} \rho A_2^- \left\{ u_1 A_1^+ + B_1 \right\} \\ \rho A_2^- \left\{ \left(\frac{p}{\rho} + u_1^2 \right) A_1^+ + u_1 B_1 \right\} \\ \rho \left\{ \left(u_1 A_1^+ + B_1 \right) \left(u_2 A_2^- - B_2 \right) \right\} \\ \rho A_2^- \left\{ u_1 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_1^+}{2} + \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_1}{2} \right\} \\ -\rho u_2 \frac{B_2}{2} \left(u_1 A_1^+ + B_1 \right) \end{bmatrix}, \quad \text{(A.4)}$$

The x-component of the split fluxes in the third quadrant are

$$\begin{bmatrix} GX^{III}(1) \\ GX^{III}(2) \\ GX^{III}(3) \\ GX^{III}(4) \end{bmatrix} = \begin{bmatrix} \rho A_2^+ \left\{ u_1 A_1^+ + B_1 \right\} \\ \rho A_2^+ \left\{ \left(\frac{p}{\rho} + u_1^2 \right) A_1^+ + u_1 B_1 \right\} \\ \rho \left\{ \left(u_1 A_1^+ + B_1 \right) \left(u_2 A_2^+ + B_2 \right) \right\} \\ \rho A_2^+ \left\{ u_1 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_1^+}{2} + \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_1}{2} \right\} \\ +\rho u_2 \frac{B_2}{2} \left(u_1 A_1^+ + B_1 \right) \end{bmatrix}, \quad \text{(A.5)}$$

The x-component of the split fluxes in the fourth quadrant are

$$\begin{bmatrix} GX^{IV}(1) \\ GX^{IV}(2) \\ GX^{IV}(3) \\ GX^{IV}(4) \end{bmatrix} = \begin{bmatrix} \rho A_2^+ \left\{ u_1 A_1^- - B_1 \right\} \\ \rho A_2^+ \left\{ \left(\frac{p}{\rho} + u_1^2 \right) A_1^- - u_1 B_1 \right\} \\ \rho \left\{ \left(u_1 A_1^- - B_1 \right) \left(u_2 A_2^+ + B_2 \right) \right\} \\ \rho A_2^+ \left\{ u_1 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_1^-}{2} - \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_1}{2} \right\} \\ +\rho u_2 \frac{B_2}{2} \left(u_1 A_1^- - B_1 \right) \end{bmatrix}, \quad \text{(A.6)}$$

The y-component of the split fluxes in the first quadrant are

$$\begin{bmatrix} GY^I(1) \\ GY^I(2) \\ GY^I(3) \\ GY^I(4) \end{bmatrix} = \begin{bmatrix} \rho A_1^- \left\{ u_2 A_2^- - B_2 \right\} \\ \rho \left\{ \left(u_1 A_1^- - B_1 \right) \left(u_2 A_2^- - B_2 \right) \right\} \\ \rho A_1^- \left\{ \left(\frac{p}{\rho} + u_2^2 \right) A_2^- - u_2 B_2 \right\} \\ \rho A_1^- \left\{ u_2 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_2^-}{2} - \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_2}{2} \right\} \\ -\rho u_1 \frac{B_1}{2} \left(u_2 A_2^- - B_2 \right) \end{bmatrix}, \quad \text{(A.7)}$$

The y-component of the split fluxes in the second quadrant are

$$\begin{bmatrix} GY^{II}(1) \\ GY^{II}(2) \\ GY^{II}(3) \\ GY^{II}(4) \end{bmatrix} = \begin{bmatrix} \rho A_1^+ \left\{ u_2 A_2^- - B_2 \right\} \\ \rho \left\{ \left(u_1 A_1^+ + B_1 \right) \left(u_2 A_2^- - B_2 \right) \right\} \\ \rho A_1^+ \left\{ \left(\frac{p}{\rho} + u_2^2 \right) A_2^- - u_2 B_2 \right\} \\ \rho A_1^+ \left\{ u_2 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_2^-}{2} - \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_2}{2} \right\} \\ +\rho u_1 \frac{B_1}{2} \left(u_2 A_2^- - B_2 \right) \end{bmatrix}, \quad \text{(A.8)}$$

The y-component of the split fluxes in the third quadrant are

$$\begin{bmatrix} GY^{III}(1) \\ GY^{III}(2) \\ GY^{III}(3) \\ GY^{III}(4) \end{bmatrix} = \begin{bmatrix} \rho A_1^+ \left\{ u_2 A_2^+ + B_2 \right\} \\ \rho \left\{ \left(u_1 A_1^+ + B_1 \right) \left(u_2 A_2^+ + B_2 \right) \right\} \\ \rho A_1^+ \left\{ \left(\frac{p}{\rho} + u_1^2 \right) A_2^+ + u_2 B_2 \right\} \\ \rho A_1^+ \left\{ u_2 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_2^+}{2} + \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_2}{2} \right\} \\ +\rho u_1 \frac{B_1}{2} \left(u_2 A_2^+ + B_2 \right) \end{bmatrix}, \quad \text{(A.9)}$$

The y-component of the split fluxes in the fourth quadrant are

$$\begin{bmatrix} GY^{IV}(1) \\ GY^{IV}(2) \\ GY^{IV}(3) \\ GY^{IV}(4) \end{bmatrix} = \begin{bmatrix} \rho A_1^- \left\{ u_2 A_2^+ + B_2 \right\} \\ \rho \left\{ \left(u_1 A_1^- - B_1 \right) \left(u_2 A_2^+ + B_2 \right) \right\} \\ \rho A_1^- \left\{ \left(\frac{p}{\rho} + u_2^2 \right) A_2^+ + u_2 B_2 \right\} \\ \rho A_1^- \left\{ u_2 \left(\frac{2\gamma}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{A_2^+}{2} + \left(\frac{\gamma+1}{\gamma-1} \frac{p}{\rho} + u_1^2 + u_2^2 \right) \frac{B_2}{2} \right\} \\ -\rho u_1 \frac{B_1}{2} \left(u_2 A_2^- + B_2 \right) \end{bmatrix}. \quad \text{(A.10)}$$

The $\bar{U}$ vector is a combination of the freestream and the value at the point P,

lying on the outer boundary surface. This is given by

$$\begin{bmatrix} \overline{U(1)} \\ \overline{U(2)} \\ \overline{U(3)} \\ \overline{U(4)} \end{bmatrix} = \begin{bmatrix} \left\{\rho A_2^-\right\}_\infty + \left\{\rho A_2^+\right\}_P \\ \left\{\rho u_1 A_2^-\right\}_\infty + \left\{\rho u_1 A_2^+\right\}_P \\ \left\{\rho\left(u_2 A_2^- - B_2\right)\right\}_\infty + \left\{\rho\left(u_2 A_2^+ + B_2\right)\right\}_P \\ \left\{\rho A_2^- e - \frac{1}{2}\rho u_2 B_2\right\}_\infty + \left\{\rho A_2^+ e + \frac{1}{2}\rho u_2 B_2\right\}_P \end{bmatrix} \tag{A.11}$$

where

$$\begin{aligned} A_1^\pm &= \frac{1 \pm \operatorname{erf} s_1}{2}, \quad A_2^\pm = \frac{1 \pm \operatorname{erf} s_2}{2}, \\ B_1 &= \frac{e^{-s_1^2}}{2\sqrt{\pi\beta}}, \quad B_2 = \frac{e^{-s_2^2}}{2\sqrt{\pi\beta}}, \quad s_1 = u_1\sqrt{\beta}, \quad s_2 = u_2\sqrt{\beta}. \end{aligned}$$

Appendix B

Split Fluxes 3-D LSKUM

The expressions for the split fluxes, in terms of the flow variables, are given below. The x-component of the flux is given by

$$\begin{bmatrix} GX^{\pm}(1) \\ GX^{\pm}(2) \\ GX^{\pm}(3) \\ GX^{\pm}(4) \\ GX^{\pm}(5) \end{bmatrix} = \begin{bmatrix} \rho\left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho\left\{\left(\frac{p}{\rho} + u_1^2\right) A_1^{\pm} \pm u_1 B_1\right\} \\ \rho u_2 \left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho u_3 \left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho\left\{u_1 \left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{A_1^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{B_1}{2}\right\} \end{bmatrix}, \tag{B.1}$$

The y-component of the flux is given by

$$\begin{bmatrix} GY^{\pm}(1) \\ GY^{\pm}(2) \\ GY^{\pm}(3) \\ GY^{\pm}(4) \\ GY^{\pm}(5) \end{bmatrix} = \begin{bmatrix} \rho\left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho u_1 \left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho\left\{\left(\frac{p}{\rho} + u_2^2\right) A_2^{\pm} \pm u_2 B_2\right\} \\ \rho u_3 \left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho\left\{u_2 \left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{A_2^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{B_2}{2}\right\} \end{bmatrix}, \tag{B.2}$$

The z-component of the flux is given by

$$\begin{bmatrix} GZ^{\pm}(1) \\ GZ^{\pm}(2) \\ GZ^{\pm}(3) \\ GZ^{\pm}(4) \\ GZ^{\pm}(5) \end{bmatrix} = \begin{bmatrix} \rho\left\{u_3 A_3^{\pm} \pm B_3\right\} \\ \rho u_1 \left\{u_3 A_3^{\pm} \pm B_3\right\} \\ \rho u_2 \left\{u_2 A_3^{\pm} \pm B_3\right\} \\ \rho\left\{\left(\frac{p}{\rho} + u_3^2\right) A_3^{\pm} \pm u_3 B_3\right\} \\ \rho\left\{u_3 \left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{A_3^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho} + q^2\right) \frac{B_2}{2}\right\} \end{bmatrix}, \tag{B.3}$$

B.1 Quadrantwise Split Fluxes for 3D-LSKUM

In case of quadrant splitting for x and y componenets of the flux, we use superscripts $+ - \cdot$ to indicate the nature of splitting. They respectively indicate positive,negative and full fluxes with the position of the superscript indicating the component. For example $GX^{+\cdot -}$ represents GX quadrant split fluxes for $v_1 > 0$, $-\infty < v_2 < \infty$ and $v_3 < 0$. The x-component of the quadrant split fluxes quadrant are

$$\begin{bmatrix} GX^{\pm\cdot\pm}(1) \\ GX^{\pm\cdot\pm}(2) \\ GX^{\pm\cdot\pm}(3) \\ GX^{\pm\cdot\pm}(4) \\ GX^{\pm\cdot\pm}(5) \end{bmatrix} = \begin{bmatrix} \rho A_3^{\pm}\left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho A_3^{\pm}\left\{\left(\frac{p}{\rho}+u_1^2\right)A_1^{\pm} \pm u_1 B_1\right\} \\ \rho A_3^{\pm} u_2\left\{u_1 A_1^{\pm} \pm B_1\right\} \\ \rho\left\{u_1 A_1^{\pm} \pm B_1\right\}\left\{u_3 A_3^{\pm} \pm B_3\right\} \\ \rho A_3^{\pm}\left\{u_1\left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho}+q^2\right)\frac{A_1^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho}+q^2\right)\frac{B_1}{2}\right\} \\ \pm\frac{1}{2}\rho u_3 B_3\left\{u_1 A_1^{\pm} \pm B_1\right\} \end{bmatrix}, \tag{B.4}$$

The y-component of the quadrant split fluxes quadrant are

$$\begin{bmatrix} GY^{\cdot\pm\pm}(1) \\ GY^{\cdot\pm\pm}(2) \\ GY^{\cdot\pm\pm}(3) \\ GY^{\cdot\pm\pm}(4) \\ GY^{\cdot\pm\pm}(5) \end{bmatrix} = \begin{bmatrix} \rho A_3^{\pm}\left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho A_3^{\pm} u_1\left\{u_2 A_2^{\pm} \pm B_2\right\} \\ \rho A_3^{\pm}\left\{\left(\frac{p}{\rho}+u_2^2\right)A_2^{\pm} \pm u_2 B_2\right\} \\ \rho\left\{u_2 A_2^{\pm} \pm B_2\right\}\left\{u_3 A_3^{\pm} \pm B_3\right\} \\ \rho A_3^{\pm}\left\{u_2\left(\frac{2\gamma}{\gamma-1}\frac{p}{\rho}+q^2\right)\frac{A_2^{\pm}}{2} \pm \left(\frac{\gamma+1}{\gamma-1}\frac{p}{\rho}+q^2\right)\frac{B_2}{2}\right\} \\ \pm\frac{1}{2}\rho u_3 B_3\left\{u_2 A_2^{\pm} \pm B_2\right\} \end{bmatrix}, \tag{B.5}$$

The $\bar{U}$ vector is a combination of the freestream and the value at the point P, lying on the outer boundary surface. This is given by

$$\begin{bmatrix} \overline{U(1)} \\ \overline{U(2)} \\ \overline{U(3)} \\ \overline{U(4)} \\ \overline{U(5)} \end{bmatrix} = \begin{bmatrix} \left\{\rho A_3^{-}\right\}_{\infty} + \left\{\rho A_3^{+}\right\}_{P} \\ \left\{\rho u_1 A_3^{-}\right\}_{\infty} + \left\{\rho u_1 A_3^{+}\right\}_{P} \\ \left\{\rho u_2 A_3^{-}\right\}_{\infty} + \left\{\rho u_2 A_3^{+}\right\}_{P} \\ \left\{\rho\left(u_3 A_3^{-} - B_3\right)\right\}_{\infty} + \left\{\rho\left(u_3 A_3^{+} + B_3\right)\right\}_{P} \\ \left\{\rho A_3^{-} e - \frac{1}{2}\rho u_3 B_3\right\}_{\infty} + \left\{\rho A_3^{+} e + \frac{1}{2}\rho u_3 B_3\right\}_{P} \end{bmatrix} \tag{B.6}$$

where

$$A_1^{\pm} = \frac{1 \pm \operatorname{erf} s_1}{2}, \quad A_2^{\pm} = \frac{1 \pm \operatorname{erf} s_2}{2}, \quad A_3^{\pm} = \frac{1 \pm \operatorname{erf} s_3}{2},$$

$$
\begin{aligned}
B_1 &= \frac{e^{-s_1^2}}{2\sqrt{\pi\beta}}, \quad B_2 = \frac{e^{-s_2^2}}{2\sqrt{\pi\beta}}, \quad B_3 = \frac{e^{-s_3^2}}{2\sqrt{\pi\beta}}, \quad q^2 = \sqrt{u_1^2 + u_2^2 + u_3^2}, \\
s_1 &= u_1\sqrt{\beta}, \quad s_2 = u_2\sqrt{\beta}, \quad s_3 = u_3\sqrt{\beta}, \quad e = \frac{RT}{\gamma - 1} + \frac{1}{2}q^2,
\end{aligned}
$$

Bibliography

[1] Albone, C. M., and Joyce, M. G., "Feature-associated mesh embedding for complex configurations", AGARD CP 464, 1990.

[2] Anderson, J.D., "Hypersonic and High Temperature Gas Dynamics", Mc-Graw-Hill Book Company(1989).

[3] AGARD-R-702, "Compendium of Unsteady Aerodynamic Measurements", Aug. 1982.

[4] AGARD-AR-211, "Test cases for inviscid flow field methods".

[5] Anandhanarayanan, K., et.al., "A KFVS based NS solver", 3rd Annual CFD Symposium, Bangalore, Aug. 11-12, 2000.

[6] Boniface, J.C., Ph. Guillen, Le Pape, M.C., Darracq, D., and Beaumier, P., "Developement of a chimera unsteady method for the numerical simulation of rotorcraft flowfields", ONERA TP 1998-4, also AIAA-98-0421.

[7] Barth, T.J., " Aspects of unstructured grids and finite volume solvers for Euler and Navier stokes quations", VKI Lecture series notes 1994-05, Brussels, 1994.

[8] Baker, T., "Mesh generation by a sequence of transformations", Appl. Num. Math., 2(1986), pp. 515-528.

[9] Benek, J., Donegan, T., and Suhs, N., " Extended Chimera grid embedding scheme with applications to viscous flows", AIAA Paper 87-1126.

[10] Clarke, D. K., Salas, M. D., and Hassan, H., "Euler calculations for multielement airfoils using Cartesian grids", AIAA J., Vol. 24, pp. 353–358, 1986.

[11] Coirier, W. J., "An adaptively-refined, Cartesian, cell-based scheme for the Euler and Navier-Stokes equations", Ph.D. Thesis, The University of Michigan, 1994.

[12] Carlson, L.A., "Transonic airfoil flow field analysis using Cartesian coordinates", NASA CR-2577 Aug. 1975.

[13] Carlson, L.A., "Transonic airfoil design using Cartesian co-ordinates", NASA CR-2587 April 1976.

[14] Carlson, L.A., " A direct -inverse method for transonic and separated flows about airfoils", NASA CR-4270.

[15] Croisille, J., and Villedieu, P., " Kinetic flux splitting schemes for hypersonic flows", in Proc 13th ICNMFD, M. Napolitano and F.Sabetta, eds., Rome, July 1992, Springer Verlag, pp. 310-313.

[16] Chakrabartty, S. K., Dhanalakshmi, K., and Mathur, J. S., "Computation of three-dimensional transonic flow using a cell vertex finite volume method for the Euler equations", Acta Mech., Vol. 115, pp. 161–177, 1996.

[17] Chakrabartty, S. K., Dhanalakshmi, K., and Mathur, J. S., "Computation of three-dimensional transonic viscous flow using the JUMBO3D code", Acta Mech., Vol. 119, pp. 181–197, 1996.

[18] Deshpande, S. M., "Kinetic theory based new upwind methods for inviscid compressible flows", AIAA Paper 86-0275, 1986.

[19] Deshpande, S. M., "On the Maxwellian distribution, symmetric form, and entropy conservation for the Euler equations", NASA TP-2583, 1986.

[20] Deshpande, S. M., "A second order accurate kinetic theory based method for inviscid compressible flow", NASA TP-2613, 1986.

[21] Deshpande, S. M., Ghosh, A. K., and Mandal, J.C., "Least squares weak upwind method for Euler equations", Fluid Mechanics Report 89 FM 4, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, 1989.

[22] Deshpande, S. M., Sekar, S., Nagarathinam, M., Krishnamurthy, R., Sinha, P. K., and Kulkarni, P. S., "A 3 dimensional upwind Euler solver using kinetic flux vector splitting method", Proc. 13th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 414, pp. 105–109, Springer, 1992.

[23] Eberle, A., " A finite volume method for calculating transonic potential flow around wings from the minimum pressure integral", NASA TM 75324, 1978.

[24] Eriksson, L., "Generation of boundary confirming grids around wing-body configurations using tranfinite interpolation", AIAA Journal, 20(1982), pp. 1313-1320.

[25] Estivalezes, J. L., and Villedieu, P, "A new second order positivity preserving kinetic scheme for the compressible Euler equations", Proc. 14th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 453, pp. 96–100, Springer, 1995.

[26] Evans, A., Marchant, M. J., Szmelter, J., and Weatherill, N. P., "Adaptivity for compressible flow computations using point embedding on 2-d structured multiblock meshes", Int. J. Numer. Methods Engg., Vol. 32, pp. 895–919, 1991.

[27] Finley, D. B., "Euler technology assessment program for preliminary aircraft design employing SPLITFLOW code with Cartesian unstructured grid method", NASA CR 4649, 1995.

[28] Finley, D. B., and Karman, S. L., "Euler technology assessment for preliminary aircraft design — compressibility predictions by employing the Cartesian unstructured grid SPLITFLOW code", NASA CR 4710, 1996.

[29] Ghosh, A. K., and Deshpande, S. M., "A robust least squares kinetic upwind scheme for Euler equations" Proc. 14th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 453, pp. 101–105, Springer, 1995.

[30] Ghosh, A. K., and Deshpande, S. M., "Least squares kinetic upwind method for inviscid compressible flows", AIAA Paper 95-1735, 1995.

[31] Ghosh, A. K., "Robust least squares kinetic upwind method for inviscid compressible flows", Ph.D. Thesis, Indian Institute of Science, Bangalore, 1996.

[32] Gillyboeuf, J.P., Mansuy, P. and Pavsic, S., " Two new chimera method : Application to missile seperation", AIAA paper 95-0353.

[33] GAMM Workshop on numerical solution of compressible Euler flows, 1986

[34] Garabedian, P.R., and Korn, D.G., " Analysis of transonic airfoils", communication on pure and applied mathematics, vol. 24, 1971, pp. 841-851.

[35] Godunov, S., " A difference method for the numerical calculation of discontinous solutions of hydrodynamic equations", Mat. Sbornik, 47(1959), pp. 271-306.

[36] Jameson, A., "Iterative solution of transonic flow over airfoils and wings, Including flows at Mach 1 ", communication on pure and applied mathematics, vol.27, 1974, pp. 283-309.

[37] Jameson, A., and Baker, T., "Improvements to the aircraft Euler method", AIAA Paper 87-0452, AIAA 35th Aerospace Sciences Meeting, Reno, NV, January 1987.

[38] Jameson, A., Schmidt, W., and Turkel, E., "Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes" AIAA Paper 81-1259, 1981.

[39] Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of inviscid transonic flow over a complete aircraft", AIAA Paper 86-0103, 1986.

[40] Jameson, A., "Essential Elements of Computational Algorithms for Aerodynamic Analysis and Design", NASA CR-97-206268.

[41] Jain Sachin B, "Least Squares Kinetic Upwind Method on Moving Grids for Unsteady Euler Computation", ME project thesis, Dept. of Aero. Engg, IISc, Bangalore, India, Jan 2000.

[42] Jain Sachin, Ramesh, V. and Deshpande, S.M., "Least squares kinetic upwind method on moving grid for unsteady Euler computations", 1st International conference on Computational Fluid Dynamics, Kyoto Japan, July 10- 14, 2000

[43] Kulkarni, P.S. and Deshpande, S.M., "KFVS on moving grid for unsteady aerodynamics", Fluid Mechanics Report 95 FM 6, Dept. of Aero. Engg., IISc, Bangalore, India, June 1995.

[44] Krishnamurthy, R., PhD thesis(under preparation), Dept. of Aero. Engg., IISc, Bangalore, India.

[45] Liepmann, H. W., and Roshko, A., "Elements of gasdynamics", Wiley, 1957.

[46] Lou,T., Dahlby,D.C. and Baganoff, D., " A numerical study comparing Flux-Vector Splitting for Navier-stokes Equations with a Particle Method", Journal of computational physics 145, 489-510(1998).

[47] Lohner, R., and Parikh, P., "Generation of three-dimensional unstructured grids by the advancing front method", AIAA Paper 72-154, 1972.

[48] Manoj Kumar,B., Raghurama Rao, S.V. and Deshpande,S.M.,*Kinetic smooth particle hydrodynamics method using peculiar velocity based upwinding and least squares*, Fluid Mechanics Report 98 FM 2, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, 1998.

[49] Mandal, J. C., "Kinetic upwind method for inviscid compressible flows", Ph.D. Thesis, Indian Institute of Science, Bangalore, 1989.

[50] Mandal, J. C., and Deshpande, S. M., "Higher order accurate kinetic flux vector splitting method for Euler equations", Proc. 2nd Int. Conf. on Nonlinear Hyperbolic Problems, in Notes on Numerical Fluid Mechanics, Vol. 24, pp. 384–392, Vieweg, 1989.

[51] Mandal, J. C., and Deshpande, S. M., "Kinetic flux vector splitting for Euler equations", Computers & Fluids, Vol. 23, pp. 447–478, 1994.

[52] Mathur, J. S., and Ramesh, V., "An upwind kinetic method for the Euler equations using adapted Cartesian grids", Proc. 1st Asian CFD Conf., Hong Kong, Vol. 1, pp. 371–376, Jan. 16–19, 1995.

[53] Mathur, J. S., and Deshpande, S. M., "Reconstruction on unstructured grids using the kinetic flux vector splitting method", Project Document NAL PD CF 9601, National Aerospace Laboratories, Bangalore, 1996.

[54] Mathur, J. S., and Deshpande, S. M., "Reconstruction on unstructured grids using an upwind kinetic method", Proc. 15th Int. Conf. on Numerical Methods in Fluid Dynamics, Monterey, California, June 24–28, 1996 (to appear).

[55] Mathur, J. S., and Ramesh, V., "Computation of inviscid compressible flow on adapted Cartesian grids using the kinetic flux vector splitting method", Comput. Fluid Dynamics J., Vol. 6, pp. 51–64, 1997.

[56] Mathur, J.S"Application of kinetic schemes to structured, unstructured,Cartesian and hybrid grids", Ph.D. Thesis, Indian Institute of Science, Bangalore, November, 1997.

[57] Mathur, J.S, and Chakrabartty, S.K., " A grid generation package for high aspect ratio wings", NAL SP 9315, pp. 115-121, National Aerospace Laboratories, Bangalore, 1993.

[58] Morinishi, K., "A finite difference solution of the Euler equations on non-body-fitted Cartesian grids" Computers & Fluids, Vol. 21, pp. 331–344, 1992.

[59] Murman, E., " Analysis of embedded shock waves calculated by relaxation methods", AIAA Journal, 12(1974), PP.626-633.

[60] Murman, E., and Cole., J., " Calculation of plane steady transonic flows", AIAA Journal, 9(1971), pp. 114-121.

[61] Paillere, P., and Deconinck, H., " A review of multi-dimensional upwind residual distribution schemes for the Euler equations", CFD review 1995, M. Hafez and K. Oshima, eds., wiley, 1995, pp. 141-160.

[62] Quirk, J. J., "An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies", Computers & Fluids, Vol. 23, pp. 125–140, 1994.

[63] Raghurama Rao, S. V., "New upwind methods based on the kinetic theory for inviscid compressible flows", Ph.D. Thesis, Indian Institute of Science, Bangalore, 1994.

[64] Raghurama Rao, S. V., and Deshpande, S. M., "Peculiar velocity based upwind method for inviscid compressible flows", Comput. Fluid Dynamics J., Vol. 3, pp. 415–432, 1995.

[65] Raghurama Rao, S. V., "Peculiar velocity based upwind method for inviscid compressible flows", Proc. 14th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 453, pp. 112–116, Springer, 1995.

[66] Ramesh, V., Mathur, J. S., and Deshpande, S. M., "Kinetic treatment of the far-field boundary condition", Fluid Mechanics Report 97 FM 2, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, 1997.

[67] Ramesh, V., Ghosh, A.K., and Deshpande, S. M., " Computation of three dimensional inviscid compressible flows using least squares kinetic upwind method ", Fluid Mechanics Report 97 FM 8, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, Sept. 1997.

[68] Ramesh, V. and Deshpande, S. M., "Least squares kinetic upwind method on moving grids", Fluid Mechanics Report 99 FM 1, Centre of Excellence in Aerospace CFD, Dept. of Aero. Engg., Indian Institute of Science, Bangalore, Jan. 1999.

[69] Ramesh,V. and Deshpande, S.M., " Least Squares Kinetic Upwind Method on moving grids for unsteady Euler computations", Computers and Fluids Journal, Vol. 30/5, pp. 621-641, May 2001.

[70] Ramesh, V., "Computation of separated flows over airfoils", M.Sc. (Engg.) Thesis, Indian Institute of Science, Bangalore, 1992.

[71] Sekar, S., Nagarathinam, M., Krishnamurthy, R., Kulkarni, P. S., and Deshpande, S. M., "3-D KFVS Euler code BHEEMA as aerodynamic design and analysis tool for complex configurations", Proc. 14th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 453, pp. 525–529, Springer, 1995.

[72] Stegar, J., and Chaussee, D., "Generation of body fitted coordinates using hyperbolic partial differential equations", SIAM J. Sci. and Stat. Comp., 1 (1980), pp. 431-437.

[73] Taylor, G.I., "The formation of a blast wave by a very intense explosion Part I. Theoretical discussions", Proc. Roy. Soc.A, 201, pp. 159–174, March 1950.

[74] Taylor, G.I.,"The formation of a blast wave by a very intense explosion Part II. The atomic explosion of 1945", Proc. Roy. Soc.A, 201, pp. 175–186, March 1950.

[75] Thompson, J., Warsi, Z., and Mastin, C., " Boundary fitted coordinate systems for numerical solution of partial differential equations: A review ", J. Comp. Phy., 47(1982), PP. 1-108

[76] VKI - Lecture series 1994-02, Grid generation, Jan 24-28, 1994.

[77] Viviand, H., "Numerical solutions of two-dimensional reference test cases", in Test cases for inviscid flow field methods, AGARD AR-211, 1985.

[78] Weatherill, N. P., and Forsey, C. R., "Grid generation and flow calculations for aircraft geometries", J. Aircraft, Vol. 22, pp. 855–860, 1985.

[79] Weatherill, N. P., "Delaunay triangulation in computational fluid dynamics", Computer Math. Applic., Vol. 24, pp. 129–150, 1992.

[80] Weatherill, N. P., Mathur, J. S., and Marchant, M. J., "An upwind kinetic flux vector splitting method on general mesh topologies", Int. J. Numer. Methods Engg., Vol. 37, pp. 623–643, 1994.

[81] Whitaker, D.L., "Three-dimensional unstructured grid Euler computations using a fully-implicit, upwind method, AIAA Paper 93-3337, 1993.

[82] Wu, Z. N., "Steady transonic flow computations using overlapping grids", Proc. 14th Int. Conf. on Numerical Methods in Fluid Dynamics, in Lecture Notes in Physics, Vol. 453, pp. 183–187, Springer, 1995.

[83] Wind tunnel experimental data(Private communication).

[84] Yerry,M.A. and Shepard, M.S., " A modified quadtree approach to finite element grid generation", IEEE G.J., p31-47, 1988.

[85] Yiu,K.F.C., Greaves,D.M., Cruz,S., Saalehi,A. and Borthwick,A.G.L., " Quadtree grid generation : Formation handling, Boundary fitting and CFD Applications", Computers & Fluids, Vol. 25, No. 8, pp. 759-769, 1996.